Dietmar Herrmann

Herrmanns Programm Sammlung

Basic CBM

für CBM 2000, 3000, 4000, 8000, VC 20

1

Spiele, Knobeleien & Simulationen

iwt

Dietmar Herrmann

Herrmanns Programm Sammlung

Basic CBM

für CBM 2000, 3000, 4000, 8000, VC 20

1

Spiele, Knobeleien & Simulationen

iwt

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Herrmann, Dietmar:
(Programmsammlung)
Herrmanns Programmsammlung / Dietmar Herrmann.
— Vaterstetten: IWT Verlag.
Basic CBM: für CBM 2000, 3000, 4000, 8000, VC 20.
1. Spiele, Knobeleien & Simulationen. — 1982.
ISBN 3-88322-013-2

ISBN 3-88322-013-2
1. Auflage 1982

Holdenrieds Druck- und Verlags-GmbH, Füssen
Umschlaggestaltung: Kaselow und Partner, München

Arbeit, Gebet, Mahl, Schlaf und Spiel
sind die fünf Finger unserer Lebenshand.
Shakespeare

Immer Arbeit, nie Spiel,
wird dem Knaben Hans zuviel.
Engl. Sprichwort

Spielen ist experimentieren
mit dem Zufall
Novalis

VORWORT

Dies ist der erste Band einer Reihe von BASIC-Programmsammlungen für den CBM-Computer. Er enthält 40 Programme, die zum Spielen, Knobeln und zur Simulation anregen sollen.

Bekannten Spielen, wie Solitaire, Knobeln, Mastermind und Nim, folgen dann interessante neue Knobeleien, die hier erstmals in BASIC für CBM vorgelegt werden wie, die vollständige Bestimmung magischer Quadrate und die Lösung von Logikaufgaben am Computer.

Für Tüftler und solche, die es werden wollen, werden Leckerbissen geboten wie: Labyrinthe mit genau einem Durchgang, die Türme von Hanoi, Springerzüge auf dem Schachbrett, das Josephus-Problem und verschiedene Logeleien.

Mit dem Zufall spielen wir beim Galtonbrett, dem Buffon-Nadelproblem oder auch dem radioaktiven Zerfall.

Zu eigenem "Forschen" können solche Probleme wie: Füchse und Hasen, Ausbreitung einer Epidemie oder auch Wasserverschmutzung anregen.

Die eckigen Klammern [ ] verweisen auf das Literaturverzeichnis am Ende des Buches.

Dem Verlag danke ich für die Herausgabe des Bandes und die stets freundliche Zusammenarbeit.

Anzing, im Juli 1982

INHALT

Seite

EINFÜHRUNG

Zur Programmiersprache BASIC

BASIC heißt Beginners All-purpose Symbolic Instruction Code und wurde 1964 von John G. Kemeny und Thomas E. Kurtz am Dartmouth College (USA) entwickelt. Zu Unterrichtszwecken erdacht, wurde BASIC mit folgenden Eigenschaften ausgestattet:
- leichte Erlernbarkeit wegen des geringen Sprachumfangs
- Dialogfähigkeit
- Maschinennähe erlaubt kompaktes Betriebsystem

Alle diese Eigenschaften führten dazu, daß fast alle Hersteller von Kleincomputern bevorzugt BASIC aufgriffen. Aber nicht nur diese, denn auch Großcomputer-Firmen wie IBM und Siemens benützen es zu Schulungszwecken. Wegen der Vielzahl der auf den Markt kommenden Systeme und wegen des schon erwähnten minimalen Sprachumfangs erfuhr BASIC so zahlreiche Erweiterungen, daß bald Hunderte von Versionen existierten. Dies ist natürlich ein entscheidendes Hindernis für die Austauschbarkeit von Software.

Zwar wurde BASIC inzwischen genormt durch die Organisationen
ECMA (European Computer Manufacturer Assoziation) 1976
ANSI (American National Standards Institute) 1978,
jedoch beschreibt dieser Standard nur ein Minimal-BASIC. Ein Blick auf die Schlüsselwort-Liste des Minimal-BASIC:

BASE DATA DEF DIM END FOR GO GOSUB GOTO IF INPUT LET NEXT ON OPTION PRINT RANDOMIZE READ REM RESTORE RETURN STEP STOP SUB THEN TO

lehrt, daß die Normierung zu spät kam und von manchen Firmen nicht akzeptiert wurde.
Der Vergleich mit den Programmiersprachen PASCAL (N. Wirth 1971) und ELAN (K. Koster/G. Hommel 1976) zeigt, daß BASIC auf einem etwas älteren Konzept beruht.
Deswegen versuchten B. Christensen und B. Loefstedt 1974 BASIC weiter zu entwickeln. Die Programmiersprache COMAL sollte an BASIC anknüpfen und trotzdem den Anforderungen des strukturierten Programmierens gerecht werden. Wie weit COMAL BASIC ablösen wird, läßt sich noch nicht sagen. Ebenso wird sich zeigen, ob sich das 1977 von K. Bowles entwickelte UCSD-PASCAL gegen BASIC durchsetzen wird.

Zum Commodore-BASIC

Wegen der erwähnten Vielzahl der BASIC-Versionen, sollen einige Eigenschaften des hier verwendeten Commodore-BASIC aufgezeigt werden:

- Der Index 0 bei Feldern ist erlaubt. Felder mit weniger als 10 Elementen müssen nicht dimensioniert werden.
- Die Schleife FOR ... TO wird stets einmal ausgeführt, auch wenn der obere Schleifenindex kleiner als der untere ist. Bei einem Sprung aus einer Schleife bleibt der Indexwert erhalten.
- Nach NEXT darf die Laufvariable fehlen.
- Nach IF .. THEN dürfen mehrere Anweisungen stehen. Sie werden jedoch nur ausgeführt, wenn die IF-Bedingung erfüllt ist.
- Nach INPUT darf ein Kommentar in Anführungszeichen stehen.
- Das Argument von RND(X) ist für x>0 beliebig; für x= –1 wird stets dieselbe Zahl ausgedrückt.
- Variablennamen dürfen mehr als 2 Buchstaben umfassen, jedoch sind nur die ersten beiden signifikant.
- Die interne Darstellung von –1 und 0 erlaubt die Wahrheitswertverknüpfungen AND, OR und NOT. Der Wahrheitswert "wahr" wird durch –1 und "falsch" durch 0 codiert.
- Die Bildschirmsteuerzeichen haben folgende Bedeutung:

```
CURSOR NACH UNTEN

CURSOR NACH OBEN

CURSOR NACH RECHTS

CURSOR NACH LINKS

CURSOR HOME

BILDSCHIRM LOESCHEN

REVERS-DRUCK

REVERS-ENDE
```

Zu den Programmen

Der Band setzt nur wenig BASIC-Kenntnisse voraus. Lesern, die eine Einführung in BASIC suchen, steht zahlreiche Literatur zur Verfügung. Besonderheiten des CBM-BASIC sind im vorangehenden Abschnitt erklärt worden.

Leider ist es so, daß sich die Lesbarkeit und Strukturiertheit von BASIC-Spielprogrammen kaum mit schneller Ausführbarkeit vereinen lassen. Manche Programme sind daher kompakter geschrieben und enthalten wenig Kommentare.

Die folgenden Programm-Versionen wurden auf CBM 3000 erarbeitet. Damit die Programme auch auf anderen Maschinen laufen, wurde mit wenigen Ausnahmen auf POKE-Befehle verzichtet. Ausnahmen sind die Programme Nr. 7 (Schiebespiel), Nr. 9 (Vier-in-einer-Reihe), Nr. 16 (Labyrinth), Nr. 34 (Radioaktiver Zerfall) und Nr. 35 (Galtonbrett).

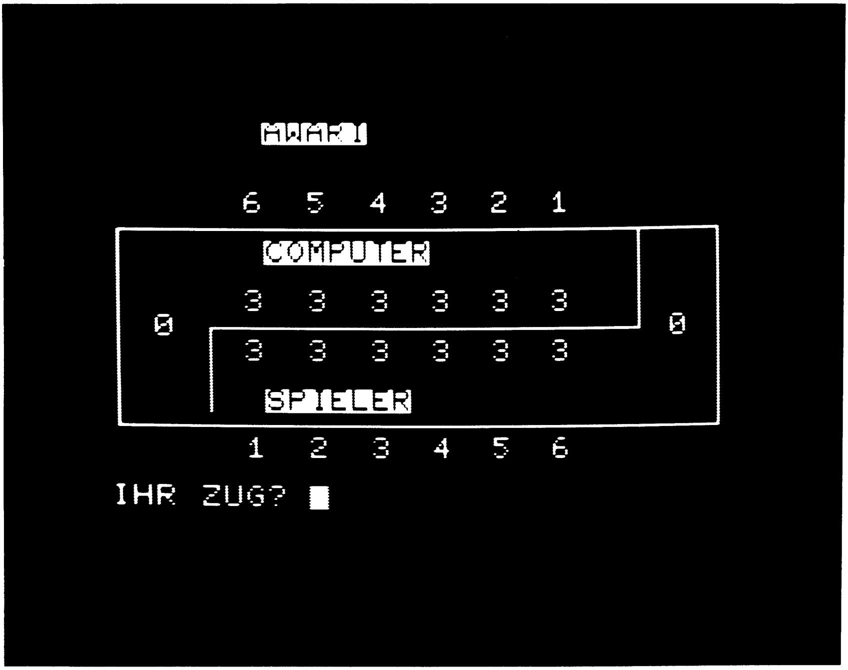
Da die POKE-Adressen für den großen Bildschirm ebenfalls mit 32768 beginnen, laufen die Programme auch auf CBM 4000 und 8000. Das einzige, kleine Maschinenprogramm beim "Radioaktiven Zerfall" dient dazu, den Bildschirm revers zu schalten. Es kann notfalls durch POKE-Anweisungen ersetzt werden.

Alle Programme ohne POKE- oder WAIT-Befehle laufen grundsätzlich auch auf dem VC-20; jedoch muß wegen des kleineren Bildschirms die Graphik etwa auf die Hälfte verkleinert werden. Eine vollständige Übertragbarkeit von Programmen auf den VC-20 ist nicht zu erreichen.
Als Muster wurden von zwei Programmen die VC-20-Version beigefügt: "Craps" und "Mastermind" (letzteres in Farbe).
Die Anfangsadresse des VC-20-Bildschirm-RAM lautet bei der 5K-Grundversion 7680, entsprechend bei der 8 K/16 K-Speichererweiterung 4096. Die Anfangsadresse des Farb-RAM verschiebt sich von 38400 in der Grundversion auf 37888 bei Aufrüstung.

1. Awari

Awari, auch unter dem Namen Kalari oder Bohnenspiel bekannt, ist ein altes afrikanisches Spiel, und wird mit Steinen, Muscheln oder eben Bohnen im Sand gespielt.

Es wird auf 14 Feldern gespielt; jeder Spieler hat vor sich sechs Spielfelder und rechts ein Ergebnisfeld.



In der hier vorgestellten Version von Awari enthält jedes der sechs Spielfelder anfangs drei Steine.

Jeder Spieler nimmt nun aus einem der Spielfelder die enthaltenen Steine heraus und legt gegen den Uhrzeigersinn jeweils einen in die folgenden Felder. Hierbei wird auch das Ergebnisfeld bedient. Die dort gesammelten Steine zählen am Schluß als Pluspunkte.

Kommt bei einem Zug der letzte zu verteilende Stein in ein eigenes Feld, so darf der Spieler noch einmal ziehen. War dieses Feld dann auch noch leer, nicht jedoch das direkt gegenüberliegende gegnerische Feld, so darf er seinen Stein und die aus diesem gegnerischen Feld in sein eigenes Ergebnisfeld legen. Das Spiel endet, wenn alle Felder 1 - 6 eines Spielers leer sind.

Zum folgenden Programm

Das obige Spielfeld wird am Bildschirm graphisch dargestellt. Der Computer übernimmt die Rolle des 2. Spielers. Der Spieler führt seine Züge einzeln durch, der Computer gleichzeitig. Es wird jeweils die neue Zahl der Steine in den einzelnen Feldern ausgegeben.

```
100 REM AWARI
110 :
120 DIM B(13),G(13),F(50)
130 PRINT"❐":E=0:N=0
140 FOR I=0 TO 12:B(I)=3:NEXT I
150 C=0:F(N)=0:B(13)=0:B(6)=0
160 GOSUB 480
170 PRINT"▮▮▮▮▮IHR ZUG";:GOSUB 320
180 IF E=0 THEN 250
190 IF M=H THEN GOSUB 310
200 IF E=0 THEN 250
210 PRINT"▮▮▮▮▮MEIN ZUG ";:GOSUB 600
220 IF E=0 THEN 250
230 IF M=H THEN PRINT",";:GOSUB 600
240 IF E>0 THEN 160
250 PRINT"▮▮▮▮▮▮▮ENDE"
260 D=B(6)-B(13)
270 IF D<0 THEN PRINT"COMPUTER GEWINNT MIT";-D;" PUNKTEN":END
280 N=N+1:IF D=0 THEN PRINT"UNENTSCHIEDEN":END
290 PRINT"SIE GEWINNEN MIT";D;" PUNKTEN":END
300 :
310 PRINT"▮▮▮▮▮NOCHEINMAL";
320 INPUT M:IF M<7 THEN IF M>0 THEN M=M-1:GOTO 340
330 PRINT"NICHT ERLAUBTER ZUG":GOTO 310
340 PRINT"❐                             ▮"
350 IF B(M)=0 THEN 330
```

```
360 H=6:GOSUB 380
370 GOTO 480
380 K=M:GOSUB 540
390 E=0:IF K>6 THEN K=K-7
400 C=C+1:IF C<9 THEN F(N)=F(N)*6+K
410 FOR I=0 TO 5:IF B(I)<>0 THEN 450
420 NEXT I
430 RETURN
440 :
450 FOR I=7 TO 12:IF B(I)<>0 THEN E=1:RETURN
460 GOTO 420
470 :
480 GOSUB 800
490 PRINT"[illegible]";
500 FOR I=12 TO 7 STEP -1:PRINTTAB((13-I)*3+2)B(I);:NEXT I:PRINT
510 PRINTTAB(1)B(13);TAB(26)B(6)
520 FOR I=0 TO 5:PRINTTAB(I*3+5)B(I);:NEXT I:PRINT
530 RETURN
540 P=B(M):B(M)=0
550 FOR P=P TO 1 STEP -1:M=M+1:IF M>13 THEN M=M-14
560 B(M)=B(M)+1:NEXT P
570 IF B(M)=1 THEN IF M<>6 THEN IF M<>13 THEN IF B(12-M)<>0 THEN 590
580 RETURN
590 B(H)=B(H)+B(12-M)+1:B(M)=0:B(12-M)=0:RETURN
600 D=-99:H=13
610 FOR I=0 TO 13:G(I)=B(I):NEXT I
```

```
620 FOR J=7 TO 12:IF B(J)=0 THEN 760
630 G=0:M=J:GOSUB 540
640 FOR I=0 TO 5:IF B(I)=0 THEN 690
650 L=B(I)+I:R=0
660 IF L>13 THEN L=L-14:R=1:GOTO 660
670 IF B(L)=0 THEN IF L<>6 THEN IF L <>13 THEN R=B(12-L)+R
680 IF R>Q THEN Q=R
690 NEXT I
700 Q=B(13)-B(6)-Q:IF C>8 THEN 740
710 K=J:IF K>6 THEN K=K-7
720 FOR I=0 TO N-1:IF 6*F(N)+K=INT(F(I)/6↑(7-C)+.1) THEN Q=Q-2
730 NEXT I
740 FOR I=0 TO 13:B(I)=G(I):NEXT I
750 IF Q>=D THEN A=J:D=Q
760 NEXT J
770 M=A:PRINTCHR$(42+M);:GOTO 380
780 FOR I=0 THEN N-1:PRINT B(I):NEXT I
790 END
800 PRINT"■■■■■■■AWARI■■"
810 A$=" ┌──────────────────────────────────────┬──────┐"
820 B$=" |                                      |      |"
830 C$=" |        ■COMPUTER■                      |      |"
840 D$=" |     ┌────────────────────────────────┘      |"
850 E$=" |     |                                       |"
860 F$=" |     |   ■SPIELER■                              |"
870 G$=" └─────────────────────────────────────────────┘"
```

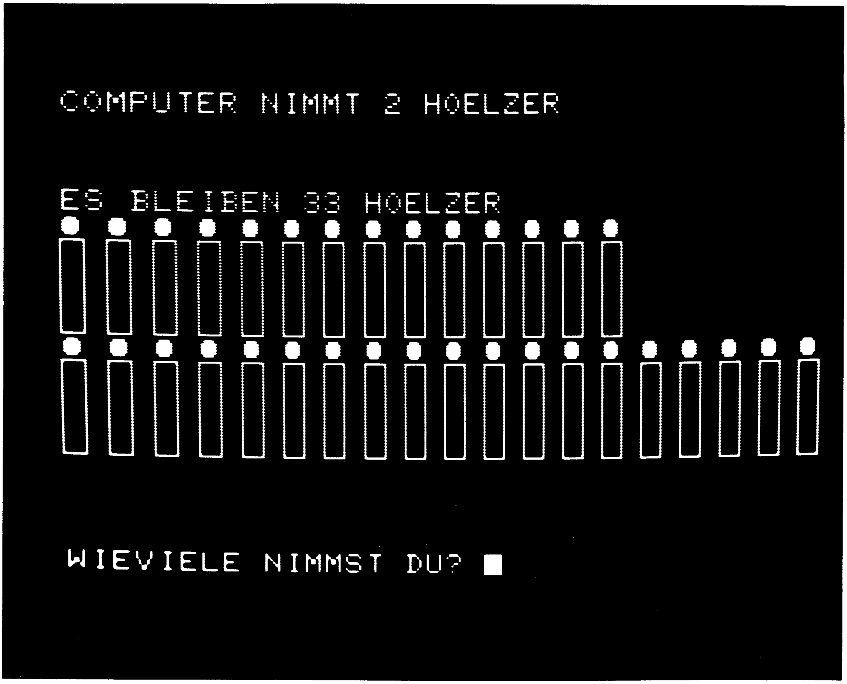
```
880 H$=" 6   5   4   3   2   1"
890 I$=" 1   2   3   4   5   6"
900 PRINTTAB(5)H$
910 PRINTA$:PRINTC$:PRINTB$:PRINTB$:PRINTD$
920 PRINTE$:PRINTE$:PRINTF$:PRINTG$
930 PRINTTAB(5)I$
940 PRINT"[illegible]"
950 RETURN
READY.
```

2. Nim

Nim ist ein seit alter Zeit bekanntes, aus China stammendes Spiel. Obwohl es sich dabei nur um die Wegnahme von Steinen, Streichhölzern o.ä. handelt, hat es doch das Interesse der Mathematiker erregt. 1901 veröffentlichte Prof. C. Bouton eine vollständige Nimspiel-Theorie in den "Annals of Mathematics"; er war es auch, der den Namen Nim erfand.

Nim wurde auf der Weltausstellung 1940 in den USA populär, als eine Maschine über 90.000 von 100.000 Spielen gewann. In Deutschland machte es 1952 Schlagzeilen, als eine Nim-Maschine, Nimrod genannt, den damaligen Wirtschaftsminister Erhard vor den Augen der Presse schlug. Auch in dem Film "Letztes Jahr in Marienbad" wurde das Spiel ausführlich dargestellt.

Es gibt natürlich zahlreiche Abwandlungen des Nimspiels: Sie unterscheiden sich entweder durch die Zahl der Haufen bzw. der maximal zu nehmenden Hölzchen oder, derjenige gewinnt oder verliert, der zuletzt zieht.



Zum folgenden Programm

Das Programm spielt das Nim mit einem Haufen mit höchstens 38 Hölzchen; wovon maximal 3 Hölzchen gezogen werden dürfen. Verlierer ist derjenige, der das letzte Hölzchen nehmen muß.

Interessant ist, daß es beim Nim eine optimale Strategie gibt, die jedoch vom ersten Zug an eingehalten werden muß. Und zwar müssen stets soviele Hölzer vom Spieler genommen werden, daß sich eine Zahl ergibt, die bei der Division durch 4 den Rest 1 läßt. Hält man diese Strategie durch, so gewinnt man stets!

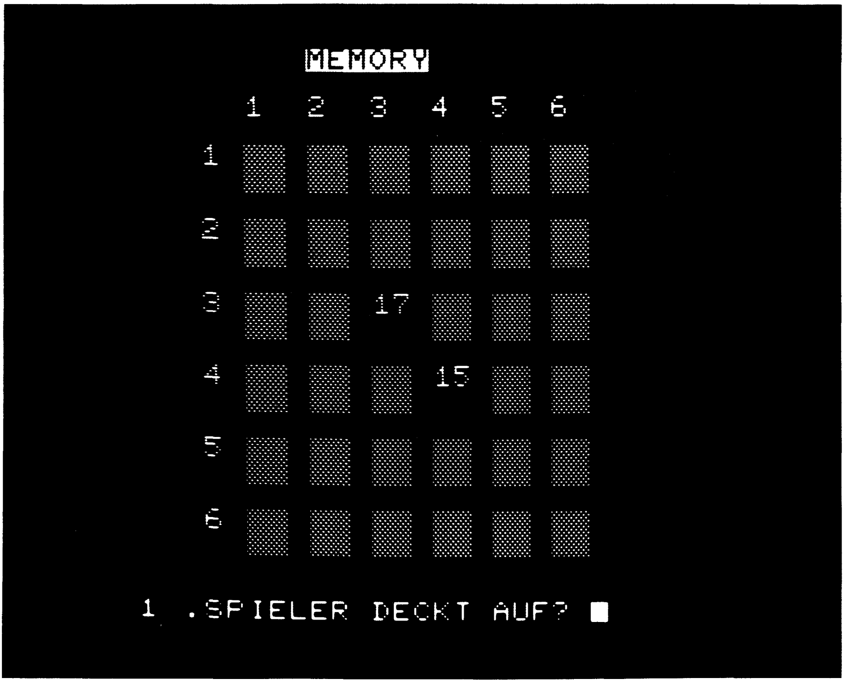
```
100 REM NIM-SPIEL
110 :
120 A$=" ┌──────────────────────────────┐"
130 B$=" |                              |"
140 C$=" |   N I M - S P I E L          |"
150 D$=" └──────────────────────────────┘"
160 FOR I=1 TO 5
170 PRINT"";TAB(5)A$
180 PRINTTAB(5)B$:PRINTTAB(5)B$:PRINTTAB(5)B$
190 PRINTTAB(5)C$
200 PRINTTAB(5)B$:PRINTTAB(5)B$:PRINTTAB(5)B$
210 PRINTTAB(5)D$
220 NEXT I
230 :
240 PRINT"SPIELREGELN"
250 PRINT""
260 PRINT"VON EINEM HAUFEN STREICHHOELZER"
270 PRINT"DARF JEDER SPIELER MIND.EINS,"
280 PRINT"HOECHSTENS JEDOCH DREI HOELZER"
290 PRINT"WEGNEHMEN.DERJENIGE VERLIERT,DER
300 PRINT"DAS LETZTE ZIEHT"
310 PRINT"WENN FERTIG,DRUECKE SHIFT"
320 WAIT 152,1
330 :
340 PRINT"SPIELER BEGINNT"
350 PRINT"WIEVIELE STREICHHOELZER SOLLEN ES SEIN  (MIN.5 MAX.38)?"
```

```
360 INPUT Z
370 IF Z>38 OR Z<5 THEN 350
380 GOSUB 660
390 :
400 PRINT"■■■":INPUT"WIEVIELE NIMMST DU";S
410 S=INT(S)
420 IF S<1 OR S>3 THEN 440
430 GOTO 450
440 PRINT"WER WIRD DENN DA MOGELN?":GOTO 400
450 Z=Z-S
460 IF Z>0 THEN 530
470 PRINT"■■■LEIDER HAST DU VERLOREN !"
480 PRINT"■■WILLST DU ES NOCHEINMAL PROBIEREN (J/N)?"
490 INPUT A$
500 IF MID$(A$,1,1)<>"J" THEN END
510 PRINT"■":GOTO 340
520 :
530 IF Z>1 THEN 570
540 PRINT"■■■GRATULIERE,DU HAST GEWONNEN !"
550 GOTO 480
560 :
570 X=Z-4*INT(Z/4)
580 IF X=1 THEN 600
590 C=X+3-4*INT((X+3)/4):GOTO 610
600 C=1
610 PRINT"■■■■■COMPUTER NIMMT";C;"HOELZER"
```

```
620 Z=Z-C
630 PRINT"[illegible]ES BLEIBEN";Z;"HOELZER"
640 GOSUB 660
650 GOTO 400
660 IF Z>19 THEN 750
670 Y=Z
680 FOR K=1 TO Y:PRINT"● ";:NEXT K:PRINT
690 FOR K=1 TO Y:PRINT"┌┐ ";:NEXT K:PRINT
700 FOR K=1 TO Y:PRINT"|| ";:NEXT K:PRINT
710 FOR K=1 TO Y:PRINT"|| ";:NEXT K:PRINT
720 FOR K=1 TO Y:PRINT"|| ";:NEXT K:PRINT
730 FOR K=1 TO Y:PRINT"└┘ ";:NEXT K:PRINT
740 RETURN
750 FOR J=1 TO Z-19:PRINT"● ";:NEXT J:PRINT
760 FOR J=1 TO Z-19:PRINT"┌┐ ";:NEXT J:PRINT
770 FOR J=1 TO Z-19:PRINT"|| ";:NEXT J:PRINT
780 FOR J=1 TO Z-19:PRINT"|| ";:NEXT J:PRINT
790 FOR J=1 TO Z-19:PRINT"|| ";:NEXT J:PRINT
800 FOR J=1 TO Z-19:PRINT"└┘ ";:NEXT J:PRINT
810 Y=19
820 GOTO 680
READY.
```

3. Memory

Memory ist ein Gedächtnisspiel, bei dem man Karten mit Zahlen oder anderen Symbolen aufdeckt und wieder verdeckt. Hat man sich zwei gleiche Karten gemerkt, so deckt man sie gleichzeitig auf und behält sie. Derjenige Spieler, der am meisten Kartenpaare aufgedeckt hat, ist Sieger.



Zum folgenden Programm

Es handelt sich um ein Zahlenmemory mit 36 Karten (18 Paare gleicher Zahlenwerte). Am Bildschirm werden die Karten in einem 6 x 6-Quadrat angeordnet. Nach Eintippen der Zeilen- und Spaltenzahl zweier Karten, werden die entsprechenden Karten aufgedeckt. Dabei wird angezeigt, ob es sich um ein Paar gleicher Zahlenwerte handelt. Ist dies nicht der Fall, so werden die Karten wieder verdeckt. Um das Spiel etwas leichter zu machen, werden zu Beginn gleich 2 Karten aufgedeckt.

```
100 REM MEMORY
110 :
120 REM SYMMETRIE FESTLEGEN
130 FOR I=1 TO 3
140 FOR K=1 TO 6
150 A(I,K)=6*(I-1)+K
160 A(I+3,K)=A(I,K)
170 NEXT K
180 NEXT I
190 REM AUSLOSEN DER NUMMERN
200 A=1
210 FOR I=1 TO 6
220 FOR K=1 TO 6
230 H=A(I,K)
240 X=INT(6*RND(I)+1)
250 Y=INT(6*RND(K)+1)
260 A(I,K)=A(X,Y):A(X,Y)=H
270 NEXT K
280 NEXT I
290 REM SPIELZUG
300 E(1)=0:E(2)=0
310 X=3:Y=3
320 U=4:V=4
330 GOSUB 490
340 FOR K=1 TO 2
350 IF K=2 THEN 370
```

```
360 U=0:V=0
370 PRINTA;".SPIELER DECKT AUF";
380 INPUT X,Y
390 IF X>6 OR Y>6 THEN 380
400 IF A(X,Y)=0 THEN 370
410 GOSUB 490
420 Z(K)=A(X,Y)
430 NEXT K
440 IF Z(1)=Z(2) THEN 740
450 PRINTTAB(30)"NIETE !"
460 A=3-A
470 GOTO 340
480 REM AUSGABE
490 PRINT"■■■■■■■■■■■MEMORY■■"
500 FOR I=1 TO 6
510 PRINTTAB(3*I+2)I;
520 NEXT I
530 PRINT:PRINT
540 F=0
550 FOR I=1 TO 6
560 PRINTTAB(3)I;
570 FOR L=1 TO 6
580 IF A(I,L)<>0 THEN 610
590 F=F+1
600 GOTO 680
610 IF I=X AND L=Y THEN 670
```

```
620 IF I=U AND L=V THEN 670
630 PRINTTAB(3*L+3)"▒▒■█▌▒▒□";
640 GOTO680
650 IF A(I,L)>=10 THEN 670
660 PRINTTAB(3*L+1)A(I,L):GOTO 680
670 PRINTTAB(3*L-2)A(I,L);
680 NEXT L:PRINT:PRINT:PRINT
690 NEXT I
700 PRINTB$
710 IF K=2 THEN 730
720 U=X:V=Y
730 RETURN
740 E(A)=E(A)+1
750 A(X,Y)=0:A(U,V)=0
760 PRINT"SPIELSTAND:"
770 PRINT"1.SPIELER:";E(1);"2.SPIELER:";E(2)
780 IF F<>24 THEN 340
790 PRINTTAB(25);"ENDE"
800 END
READY.
```

4. Mastermind

Mastermind wurde von seinem Erfinder M. Meirovich auf der Nürnberger Spielwaren-Messe 1971 vorgestellt. Es wurde dann in den USA bekannt und kam von dort wieder nach Deutschland. Auch von Mastermind existieren zahlreiche Versionen.

Die hier vorgestellte ist folgende:
Der Spieler muß die Farbkombination von 4 Steinen erraten, die zufällig aus 6 verschiedenen Farben ausgewählt wurden; dabei dürfen sich jedoch die Farben wiederholen. Die Farben sind:

rot, orange, gelb, grün, blau und purpur.

Ein Spiel besteht aus 10 Rateversuchen; bei jedem Versuch erhält der Spieler eine gewisse Anzahl weißer und schwarzer Kugeln. Die Anzahl der schwarzen Kugeln gibt an, wieviele Steine seines Rateversuchs nach Farbe und Position richtig bestimmt worden sind. Entsprechend gibt die Zahl der weißen Kugeln an, von wievielen Steinen nur die Farbe richtig geraten worden ist. Der Spieler gewinnt, wenn es ihm gelingt, durch systematisches Raten innerhalb von 10 Versuchen den richtigen Farb-Code zu bestimmen.

Zum folgenden Programm

Um Tipparbeit zu sparen, werden die 6 Farben wie folgt abgekürzt:

R = Rot	O = Orange	Y = Gelb (Yellow)
B = Blau	G = Grün	P = Purpur

Am Bildschirm wird die jeweilige Zahl von schwarzen bzw. weißen Kugeln angezeigt; ebenso die Nummer des Rateversuchs.
Durch Eingabe von "Spiel" erhält der Spieler den Spielverlauf nocheinmal ausgedruckt. Durch Eingabe von "Ende" wird das Spiel unterbrochen. Das Spiel endet mit der Bekanntgabe des richtigen Farbecodes.

```
100 REM MASTERMIND
110 :
120 PRINT"▮"
130 PRINTTAB(8)"┌─────────────────────┐"
140 PRINTTAB(8)"│                     │"
150 PRINTTAB(8)"│M A S T E R M I N D │"
160 PRINTTAB(8)"│                     │"
170 PRINTTAB(8)"└─────────────────────┘"
180 PRINT"▮▮▮SPIELREGELN BEKANNT (J/N)":INPUT A$
190 IF MID$(A$,1,1)="J" THEN 290
200 PRINT"ES MUSS DIE REIHENFOLGE EINES VIER-"
210 PRINT"STELLIGEN FARBCODES ERRATEN,WOBEI"
220 PRINT"JEDE FARBE AUCH MEHRFACH VORKOMMEN"
230 PRINT"KANN.FUER JEDE RICHTIGE FARBE ERHAELT"
240 PRINT"DER SPIELER EINEN WEISSEN,FUER JEDE"
250 PRINT"FARBE AN DER RICHTIGEN POSITION"
260 PRINT"EINEN SCHWARZEN STEIN.FERTIG=> ▮RETURN"
270 GET A$:IF A$="" THEN 270
280 PRINT"▮▮▮▮▮▮▮▮▮";
290 PRINT"DIE FARBEN SIND"
300 PRINT"▮▮▮▮▮▮R=▮ROT▮  O=▮ORANGE▮  Y=▮GELB▮"
310 PRINT"▮▮▮▮▮▮B=▮BLAU▮ G=▮GRUEN▮ P=▮PURPUR▮"
320 DIM B$(10),Y(10),Z(10),F(6)
330 F(0)=4
340 FOR I=1 TO 4
350 F(I)=INT(6*RND(1)+1)
```

```
360 NEXT I
370 FOR I=1 TO 4
380 X=F(I)
390 GOSUB 920
400 F(I)=X
410 NEXT I
420 P$=""
430 FOR K=1 TO 4
440 P$=P$+CHR$(F(K))
450 NEXT K
460 FOR P=1 TO 10
470 PRINT"██"
480 PRINTP;"█.TER ZUG(SPIEL/ENDE)";
490 INPUT G$:PRINT
500 IF G$="SPIEL" THEN 1000
510 IF G$="ENDE" THEN 620
520 B$(P)=G$
530 GOSUB 670
540 IF B=4 THEN 1070
550 GOSUB 770
560 PRINTB;"SCHWARZE STEINE"
570 Y(P)=B
580 PRINTW;"WEISSE STEINE"
590 Z(P)=W
600 NEXT P
```

```
610 PRINT"LEIDER HAST DU VERLOREN!"
620 PRINT"DER RICHTIGE CODE WAR",P$
630 PRINT"":INPUT"NOCHEINMAL";A$
640 IF MID$(A$,1,1)="J" THEN 340
650 END
660 REM BERECHNUNG DER SCHWARZEN STEINE
670 FOR K=1 TO 4
680 G(K)=ASC(MID$(G$,K,1))
690 NEXT K
700 B=0
710 FOR K=1 TO 4
720 IF G(K)<>F(K) THEN 740
730 B=B+1
740 NEXT K
750 RETURN
760 REM BERECHNUNG DER WEISSEN STEINE
770 FOR K=1 TO 4
780 R(K)=ASC(MID$(P$,K,1))
790 NEXT K
800 W=0
810 FOR I=1 TO 4
820 FOR J=1 TO 4
830 IF G(I)<>R(J) THEN 870
840 W=W+1
850 R(J)=0
860 GOTO 880
```

```
870 NEXT J
880 NEXT I
890 W=W-B
900 RETURN
910 REM CODIEREN
920 ON X GOTO 930,940,950,960,970,980
930 X=89:RETURN
940 X=82:RETURN
950 X=80:RETURN
960 X=79:RETURN
970 X=71:RETURN
980 X=66:RETURN
990 REM ERGEBNIS
1000 V=P-1
1010 PRINT"VERSUCH","SCHWARZ","WEISS"
1020 PRINT"————————","————————","——————"
1030 FOR I=1 TO V
1040 PRINTB$(I),Y(I),Z(I)
1050 NEXT I
1060 GOTO 470
1070 PRINT"DU HAST GEWONNEN!"
1080 GOTO 630
1090 END
READY.
```

```
100 REM MASTERMIND VERSION FUER VC 20
110 :
120 PRINT"█"
130 PRINT TAB(0)"┌───────────────────┐"
140 PRINT TAB(0)"│                   │"
150 PRINT TAB(0)"│M A S T E R M I N D│"
160 PRINT TAB(0)"│                   │"
170 PRINT TAB(0)"└───────────────────┘"
180 PRINT"███SPIELREGELN BEKANNT   (J/N)":INPUT A$
190 IF MID$(A$,1,1)="J" THEN 270
200 PRINT"ES MUSS DIE REIHENFOL-GE EINES VIERSTELLIGEN"
210 PRINT"█FARBCODES ERRATEN WER-DEN, WOBEI JEDE FARBE"
220 PRINT"AUCH MEHRFACH VORKOM- MEN KANN. FUER JEDE"
230 PRINT"RICHTIGE FARBE ERHAELTDER SPIELER EINEN"
240 PRINT"WEISSEN, FUER JEDE    FARBE AN DER RICHTIGEN"
250 PRINT"█POSITION EINEN SCHWAR-ZEN STEIN. =>█RETURN■"
260 GET A$:IFA$="" THEN 260
270 PRINT"█DIE FARBEN, DIE DURCH DRUECKEN DER VC20FARB-TASTEN
280 PRINT"██ ROT ■  ██TUERKIS■  ██ GELB ■█"  WAEHLBAR SIND:"
290 PRINT"██ BLAU ■  ██GRUEN■  ██PURPUR■█"
300 DIM B$(10),Y(10),Z(10),F(6)
310 F(0)=4
320 FORI=1 TO 4
330 F(I)=INT(6*RND(1)+1)
340 NEXT I
350 FOR I=1 TO 4
```

```
360 X=F(I)
370 GOSUB 1070
380 F(I)=X
390 NEXT I
400 P$=""
410 FOR K=1 TO 4
420 P$=P$+CHR$(F(K))
430 NEXT K
440 FOR P=1 TO 10
450 PRINT"■■■"
460 PRINT P;"■.TER ZUG(SPIEL/ENDE)"
470 G$=""
480 FOR T=1 TO 4
490 GET L$:IF L$="" THEN 490
500 IF L$="3" THEN PRINT"■ ■";:GOTO580
510 IF L$="4" THEN PRINT"■ ■";:GOTO580
520 IF L$="5" THEN PRINT"■ ■";:GOTO580
530 IF L$="6" THEN PRINT"■ ■";:GOTO580
540 IF L$="7" THEN PRINT"■ ■";:GOTO580
550 IF L$="8" THEN PRINT"■ ■";:GOTO580
560 IF L$="S" THEN PRINT"■SPIEL":GOTO 1150
570 IF L$="E" THEN PRINT"■ENDE":GOTO 690
580 G$=G$+L$:PRINT"■";:NEXT T
590 B$(P)=G$
600 GOSUB 820
610 IF B=4 THEN 1300
```

```
620 GOSUB 920
630 PRINT B;"SCHWARZE STEINE"
640 Y(P)=B
650 PRINT"■■■■";W;"WEISSE STEINE"
660 Z(P)=W
670 NEXT P
680 PRINT"■■■■     LEIDER HAST DU              VERLOREN!"
690 PRINT"DER RICHTIGE CODE WAR"
700 FOR R=1 TO 4
710 IF MID$(P$,R,1)="3" THEN PRINT"■■ ■ ";:GOTO770
720 IF MID$(P$,R,1)="4" THEN PRINT"■■ ■ ";:GOTO770
730 IF MID$(P$,R,1)="5" THEN PRINT"■■ ■ ";:GOTO 770
740 IF MID$(P$,R,1)="6" THEN PRINT"■■ ■ ";:GOTO 770
750 IF MID$(P$,R,1)="7" THEN PRINT"■■ ■ ";:GOTO 770
760 IF MID$(P$,R,1)="8" THEN PRINT"■■ ■ ";
770 PRINT"■";:NEXT R
780 P$="":PRINT"■■■":INPUT"NOCH EINMAL";A$
790 IF MID$(A$,1,1)="J" THEN 320
800 END
810 REM BERECHNUNG DER    SCHWARZEN STEINE
820 FOR K=1 TO 4
830 G(K)=ASC(MID$(G$,K,1))
840 NEXT K
850 B=0
860 FOR K=1 TO 4
870 IFG(K)<>F(K) THEN 890
```

```
880 B=B+1
890 NEXT K
900 RETURN
910 REM BERECHNUNG DER     WEISSEN STEINE
920 FOR K=1 TO4
930 R(K)=ASC(MID$(P$,K,1))
940 NEXT K
950 W=0
960 FOR I=1 TO 4
970 FOR J=1 TO 4
980 IF G(I)<>R(J) THEN 1020
990 W=W+1
1000 R(J)=0
1010 GOTO 1030
1020 NEXT J
1030 NEXT I
1040 W=W-B
1050 RETURN
1060 REM CODIEREN
1070 ON X GOTO 1080,1090,1100,1110,1120,1130
1080 X=51:RETURN
1090 X=52:RETURN
1100 X=56:RETURN
1110 X=55:RETURN
1120 X=54:RETURN
1130 X=53:RETURN
```

```
1140 REM ERGEBNIS
1150 V=P-1
1160 PRINT"VERSUCH SCHWARZ WEISS"
1170 PRINT"——————— ——————— —————"
1180 FORI=1 TO V
1190 FORX=1 TO 4
1200 IF MID$(B$(I),X,1)="3" THEN PRINT"▚▚ ▀ ";:GOTO1260
1210 IF MID$(B$(I),X,1)="4" THEN PRINT"◣▚ ▀ ";:GOTO1260
1220 IF MID$(B$(I),X,1)="5" THEN PRINT"▒▚ ▀ ";:GOTO1260
1230 IF MID$(B$(I),X,1)="6" THEN PRINT"▞▚ ▀ ";:GOTO1260
1240 IF MID$(B$(I),X,1)="7" THEN PRINT"▄▚ ▀ ";:GOTO1260
1250 IF MID$(B$(I),X,1)="8" THEN PRINT"▛▚ ▀ ";
1260 PRINT"▀";:NEXT X
1270 PRINT"▌";Y(I);"▐▐▐▐▐";Z(I)
1280 NEXT I
1290 GOTO 450
1300 PRINT"   DU HAST GEWONNEN!"
1310 GOTO780
1320 END
READY.
```

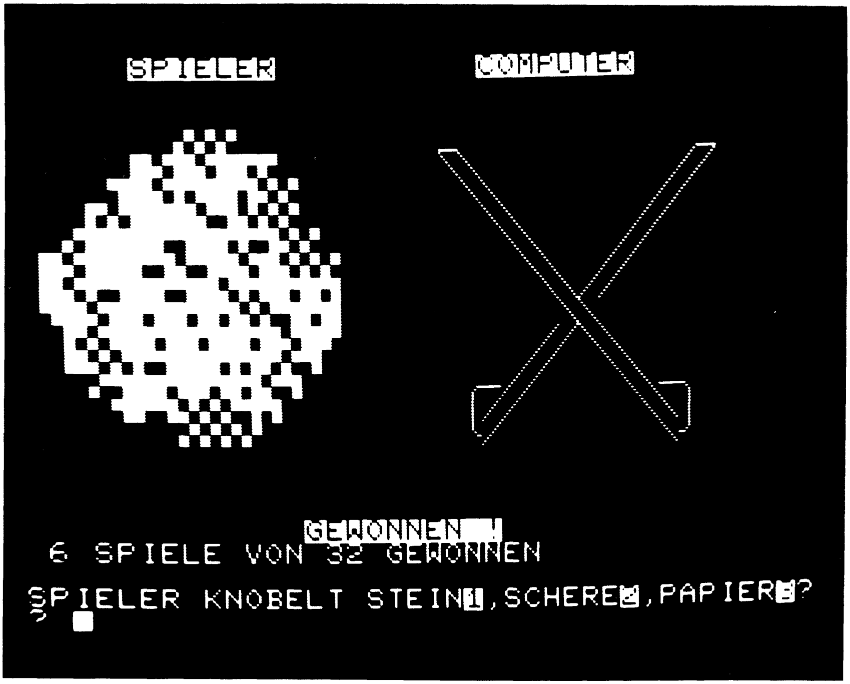
5. Knobeln

Knobeln (auch Schere-Stein-Papier) genannt, ist ein Fingerspiel, bei dem beide Spieler gleichzeitig eine der folgenden Handbewegungen machen: flache Hand (bedeutet Papier), 2 Finger spreizen (bedeutet Schere) oder Faust ballen (bedeutet Stein).

Spielregel ist

Papier schlägt Stein	(Papier rollt Stein ein)
Stein schlägt Schere	(Stein zertrümmert Schere)
Schere schlägt Papier	(Schere schneidet Papier),

bei gleichen Spielzügen gilt Unentschieden. Es gibt keine optimale Strategie, da man bei jedem Spielzug geschlagen werden kann.



Zum folgenden Programm

Nach Eingabe von 1 für Stein bzw. 2 für Schere oder 3 für Papier, werden die entsprechenden Symbole am Bildschirm gleichzeitig dargestellt. Ein Zähler hält die Anzahl der gewonnenen Spiele fest.

```
100 REM KNOBELN
110 :
120 Z=0:G=0
130 PRINT"SPIELER KNOBELT STEIN 1 ,SCHERE 2 ,PAPIER 3 ?"
140 INPUT A:Z=Z+1
150 PRINT""
160 IF A<>1 AND A<>2 AND A<>3 THEN 130
170 PRINT"SPIELER                COMPUTER"
180 X=0:PRINT""
190 ON A GOSUB 280,430,580
200 B=INT(3*RND(1)+1)
210 X=20:PRINT""
220 ON B GOSUB 280,430,580
230 PRINT"";
240 IF A=B THEN PRINT"UNENTSCHIEDEN":GOTO 270
250 IF B-A=1 OR (A=3 AND B=1) THEN PRINT"GEWONNEN !":G=G+1:GOTO 270
260 PRINT"LEIDER VERLOREN!"
270 PRINTG;"SPIELE VON";Z;"GEWONNEN":GOTO 130
280 REM STEIN
290 PRINTTAB(X)"                "
300 PRINTTAB(X)"                "
310 PRINTTAB(X)"                "
320 PRINTTAB(X)"                "
330 PRINTTAB(X)"                "
340 PRINTTAB(X)"                "
350 PRINTTAB(X)"                "
```

```
360 PRINTTAB(X)" ■■ ■■ ■ ■ ■■■■■ ■ "
370 PRINTTAB(X)"  ■ ■■  ■  ■   ■ ■  ■"
380 PRINTTAB(X)"  ■ ■■■ ■■   ■ ■ ■■■ ■"
390 PRINTTAB(X)"   ■■   ■  ■ ■ ■■ ■ "
400 PRINTTAB(X)"    ■■  ■■■■■■ ■ ■  "
410 PRINTTAB(X)"       ■■■■■■■    "
420 RETURN
430 REM SCHERE
440 PRINTTAB(X)"_             _  "
450 PRINTTAB(X)"\\            //  "
460 PRINTTAB(X)" \\          //   "
470 PRINTTAB(X)"  \\        //    "
480 PRINTTAB(X)"   \\      //     "
490 PRINTTAB(X)"    \\    //      "
500 PRINTTAB(X)"     \\\//        "
510 PRINTTAB(X)"      \\          "
520 PRINTTAB(X)"     //^\\        "
530 PRINTTAB(X)"    //    \\      "
540 PRINTTAB(X)" ,-//      \\-,   "
550 PRINTTAB(X)" |//        \\|   "
560 PRINTTAB(X)" \/          \/   "
570 RETURN
580 REM PAPIER
590 PRINTTAB(X)"┌─────────────┐     "
600 PRINTTAB(X)"|             |     "
610 PRINTTAB(X)"|             |     "
```

```
620 PRINTTAB(X)"|                |     "
630 PRINTTAB(X)"|   PAPIER       |     "
640 PRINTTAB(X)"|                |     "
650 PRINTTAB(X)"|                |     "
660 PRINTTAB(X)"|                |     "
670 PRINTTAB(X)"|                |     "
680 PRINTTAB(X)"|                |     "
690 PRINTTAB(X)"|                |     "
700 PRINTTAB(X)"|                |     "
710 PRINTTAB(X)"|________________|     "
720 RETURN
READY.
```

6. Craps

Craps ist ein in den USA beliebtes Spiel mit zwei Würfeln, das auch in Spielcasinos angeboten wird.

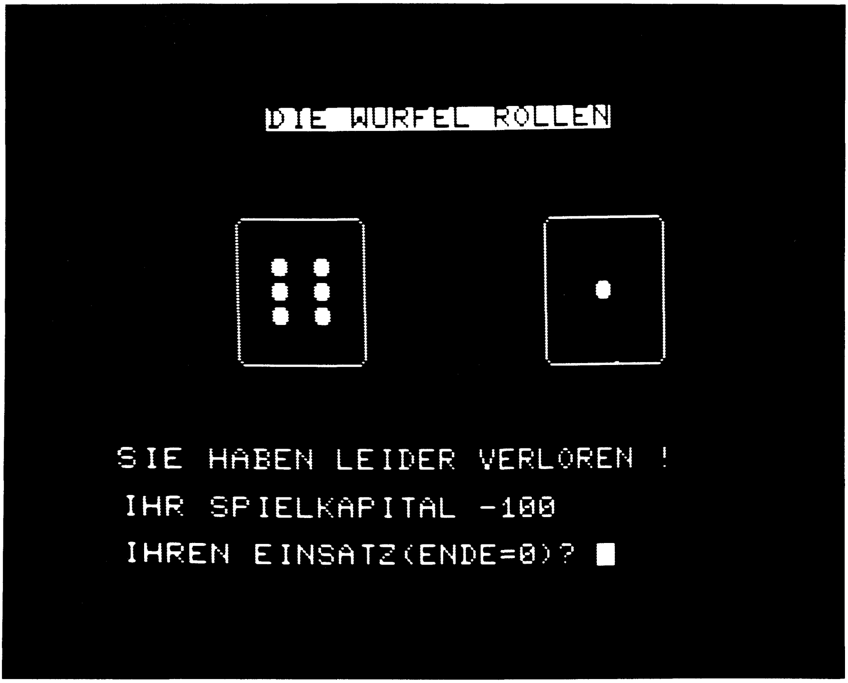
Die Regeln lauten: Zeigen beide Würfel die Augensummen

7 oder 11 beim ersten Wurf,
so gewinnt der Spieler.

2, 3 oder 12
so verliert der Spieler.

Tritt keine der beiden Möglichkeiten ein, wird das Würfeln solange fortgesetzt, bis entweder die zuerst gewürfelte Summenzahl noch einmal erscheint –
der Spieler hat gewonnen,

oder die 7 fällt –
diesmal hat der Spieler verloren!



Zum folgenden Programm

Am Bildschirm wird das Würfeln graphisch dargestellt, selbständig wiederholt und Gewinn und Verlust festgestellt. Entsprechend wird der Einsatz verrechnet.

```
100 REM CRAPS-SPIEL
110 :
120 A$=" ┌───────┐"
130 B$=" │       │"
140 C$=" └───────┘"
150 D$=" │●      │"
160 E$=" │ ●   ● │"
170 F$=" │   ●   │"
180 G$=" │      ●│"
190 FOR J=1 TO 25
200 PRINT"CRAPS"
210 PRINT"CRAPS"
220 NEXT J
230 :
240 INPUT"SPIELREGELN BEKANNT (J/N)";S$
250 IF MID$(S$,1,1)="N"THEN 270
260 GOTO 370
270 PRINT"DIE SPIELREGELN"
280 PRINT"ES WIRD MIT ZWEI WUERFELN GEWUERFELT:"
290 PRINT"AUGENSUMME 7 UND 11 GEWINNT"
300 PRINT"AUGENSUMME 2,3 UND 12 VERLIERT"
310 PRINT"BEI ALLEN ANDEREN AUGENSUMMEN WIRD NOCHEINMAL GEWUERFELT "
320 PRINT"MAN GEWINNT,WENN MAN DIESELBE AUGENSUMME WIEDER ERHAELT"
330 PRINT"BEI 7 IST DAS SPIEL VERLOREN !"
340 PRINT"WENN FERTIG,DANN AUF SHIFT DRUECKEN"
350 WAIT 152,1
```

```
360 :
370 PRINT"■":INPUT"IHREN EINSATZ";E:E=INT(E)
380 IF E<=0 THEN 370
390 GOSUB 560
400 IF S=7 OR S=11 THEN PRINT"SIE HABEN GEWONNEN !":G=G+E:GOTO 490
410 IF S=2 OR S=3 OR S=12 THEN PRINT"SIE HABEN VERLOREN !":G=G+E:GOTO
                                                                  490
420 PRINT S;" IST IHRE CHANCE!"
430 S1=S
440 FOR I=1 TO 750:NEXT I
450 GOSUB 560
460 IF S=7 THEN PRINT"SIE HABEN LEIDER VERLOREN !":G=G-E:GOTO 490
470 IF S=S1 THEN PRINT"SIE HABEN GEWONNEN !":G=G+E:GOTO 490
480 PRINT"NOCH EINMAL !":GOTO 440
490 FOR I=1 TO 750:NEXT I
500 PRINT"■IHR SPIELKAPITAL ";G
510 PRINT:INPUT"IHREN EINSATZ(ENDE=0)";E
520 IF E=0 THEN END
530 IF E< 0 THEN PRINT"NICHT MOGELN !":GOTO 510
540 GOTO 390
550 END
560 PRINT"■■■■■■■■■■■DIE WURFEL ROLLEN■"
570 FOR I=1 TO 6
580 A=INT(6*RND(I)+1):P=5:S=A
590 GOSUB 660
600 A=INT(6*RND(I)+1):P=20:S=S+A
610 GOSUB 660
```

```
 620 NEXT I
 630 PRINT"■■"
 640 RETURN
 650 :
 660 REM WUERFEL
 670 ON A GOTO 680,690,700,710,720,730
 680 A$(1)=B$:A$(2)=B$:A$(3)=F$:A$(4)=B$:A$(5)=B$:GOTO 740
 690 A$(1)=B$:A$(2)=D$:A$(3)=B$:A$(4)=G$:A$(5)=B$:GOTO 740
 700 A$(1)=B$:A$(2)=D$:A$(3)=F$:A$(4)=G$:A$(5)=B$:GOTO 740
 710 A$(1)=B$:A$(2)=E$:A$(3)=B$:A$(4)=E$:A$(5)=B$:GOTO 740
 720 A$(1)=B$:A$(2)=E$:A$(3)=F$:A$(4)=E$:A$(5)=B$:GOTO 740
 730 A$(1)=B$:A$(2)=E$:A$(3)=E$:A$(4)=E$:A$(5)=B$
 740 PRINT"■■■■■■■■";TAB(P)A$
 750 FOR K=1 TO 5
 760 PRINTTAB(P)A$(K)
 770 NEXT K
 780 PRINTTAB(P)C$
 790 RETURN
READY.
```

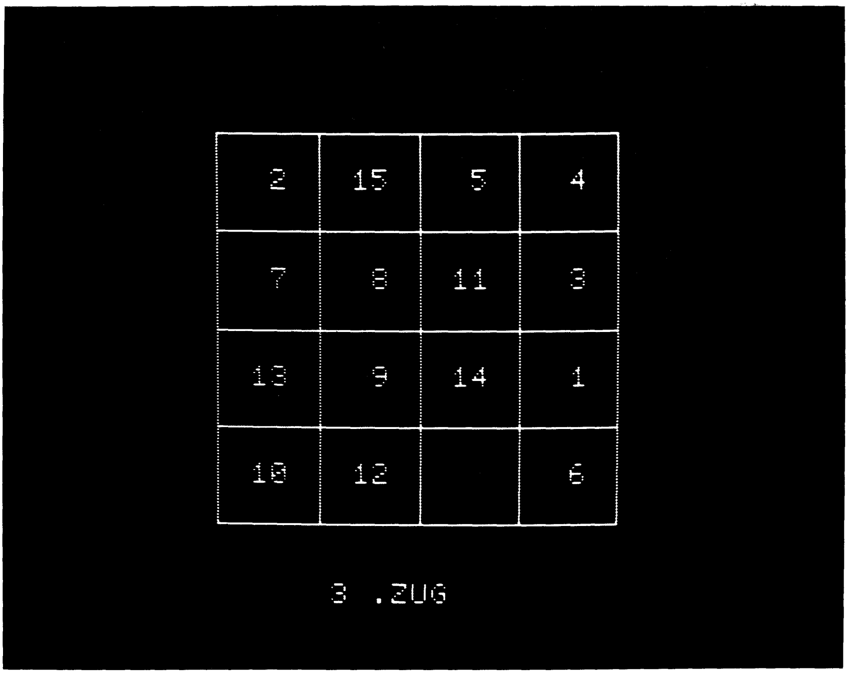
```
100 REM CRAPS-SPIEL VERSION VC-20
110 :
120 A$=" ╭──────╮"
130 B$="│      │"
140 C$=" ╰──────╯"
150 D$="│●     │"
160 E$="│ ● ●  │"
170 F$="│  ●   │"
180 G$="│    ●│"
190 FORJ=1 TO 25
200 PRINT"█████████████CRAPS"
210 PRINT"██████████████CRAPS███"
220 NEXT J
230 :
240 PRINT"SPIELREGELN BEKANNT   (J/N)":INPUT S$
250 IF MID$(S$,1,1)="N" THEN 270
260 GOTO 370
270 PRINT"████████DIE SPIELREGELN██"
280 PRINT"ES WIRD MIT ZWEI WUER-FELN GEWUERFELT:"
290 PRINT"█AUGENSUMME █7█ UND █11█   GEWINNT."
300 PRINT"AUGENSUMME █2█, █3█ UND █12█VERLIERT."
310 PRINT"█BEI ALLEN ANDEREN      AUGENSUMMEN WIRD NOCH EINMAL GEWUERFELT."
320 PRINT"█MAN GEWINNT, WENN MAN DIESELBE AUGENSUMME   WIEDER ERHAELT."
330 PRINT"█BEI █7█ IST DAS SPIEL   VERLOREN !"
340 PRINT"WENN FERTIG, DANN AUF █RETURN█ DRUECKEN"
350 GET A$:IFA$="" THEN 350
```

```
360
370 PRINT"█":INPUT"IHREN EINSATZ";E:E=INT(E)
380 IF E<=0 THEN 370
390 GOSUB 560
400 IF S=7 OR S=11 THEN PRINT"SIE HABEN GEWONNEN !":G=G+E:GOTO 490
410 IF S=2 OR S=3 OR S=12 THEN PRINT"SIE HABEN VERLOREN !":G=G+E:GOTO 490
420 PRINT S;" IST IHRE CHANCE!"
430 S1=S
440 FOR I=1 TO 750:NEXT I
450 GOSUB 560
460 IF S=7 THEN PRINT"SIE HABEN LEIDER       VERLOREN !":G=G-E:GOTO490
470 IF S=S1 THEN PRINT"SIE HABEN GEWONNEN !":G=G+E:GOTO490
480 PRINT"NOCH EINMAL !":GOTO 440
490 FOR I=1 TO 750:NEXT I
500 PRINT"█IHR SPIELKAPITAL ";G
510 PRINT:INPUT"IHR EINSATZ (ENDE=0)";E
520 IF E=0 THEN END
530 IF E<=0 THEN PRINT"NICHT MOGELN !":GOTO 510
540 GOTO 390
550 END
560 PRINT"██████DIE WUERFEL ROLLEN█"
570 FOR I=1 TO 6
580 A=INT(6*RND(I)+1):P=2:S=A
590 GOSUB 660
600 A=INT(6*RND(I)+1):P=13:S=S+A
```

```
610 GOSUB 660
620 NEXT I
630 PRINT"■■"
640 RETURN
650
660 REM WUERFEL
670 ON A GOTO 680,690,700,710,720,730
680 A$(1)=B$:A$(2)=B$:A$(3)=F$:A$(4)=B$:A$(5)=B$:GOTO 740
690 A$(1)=B$:A$(2)=D$:A$(3)=B$:A$(4)=G$:A$(5)=B$:GOTO 740
700 A$(1)=B$:A$(2)=D$:A$(3)=F$:A$(4)=G$:A$(5)=B$:GOTO 740
710 A$(1)=B$:A$(2)=E$:A$(3)=B$:A$(4)=E$:A$(5)=B$:GOTO 740
720 A$(1)=B$:A$(2)=E$:A$(3)=F$:A$(4)=E$:A$(5)=B$:GOTO 740
730 A$(1)=B$:A$(2)=E$:A$(3)=E$:A$(4)=E$:A$(5)=B$
740 PRINT"■■■■■■■";TAB(P)A$
750 FOR K=1 TO 5
760 PRINT TAB(P)A$(K)
770 NEXT K
780 PRINT TAB(P)C$
790 RETURN
READY.
```

7. Schiebespiel

Das auch heute noch in Kaufhäusern erhältliche Schiebespiel wurde um 1870 von dem wohl größten amerikanischen Rätselerfinder Sam Loyd auf den Markt gebracht.



Mit einem Trick machte er das Spiel schlagartig populär. Er versprach nämlich demjenigen 1000 Dollar Belohnung, der es schaffen würde, das Schiebespiel in die richtige Reihenfolge zu bringen. Tausende behaupteten, das Problem gelöst zu haben, keiner konnte jedoch seinen Lösungsweg angeben.

Tatsächlich aber ist das Problem unlösbar. Man kann zeigen, daß man nach dem Vertauschen zweier benachbarter Zahlen – wie 14 und 15 – auch durch beliebiges Verschieben die natürliche Ordnung nicht mehr wiederherstellen kann.

Zum folgenden Programm

Im Programm wird eine zufällige Anordnung der Zahlen 1 bis 15 erzeugt, die eine Lösung in aufsteigender Ordnung zuläßt. Mit Hilfe der numerischen Tastatur können die Felder nach Wunsch verschoben werden: Drücken der "2" bedeutet Verschieben nach unten; dem numerischen Tastenfeld entsprechend "6" nach rechts, "8" nach oben und "4" nach links.

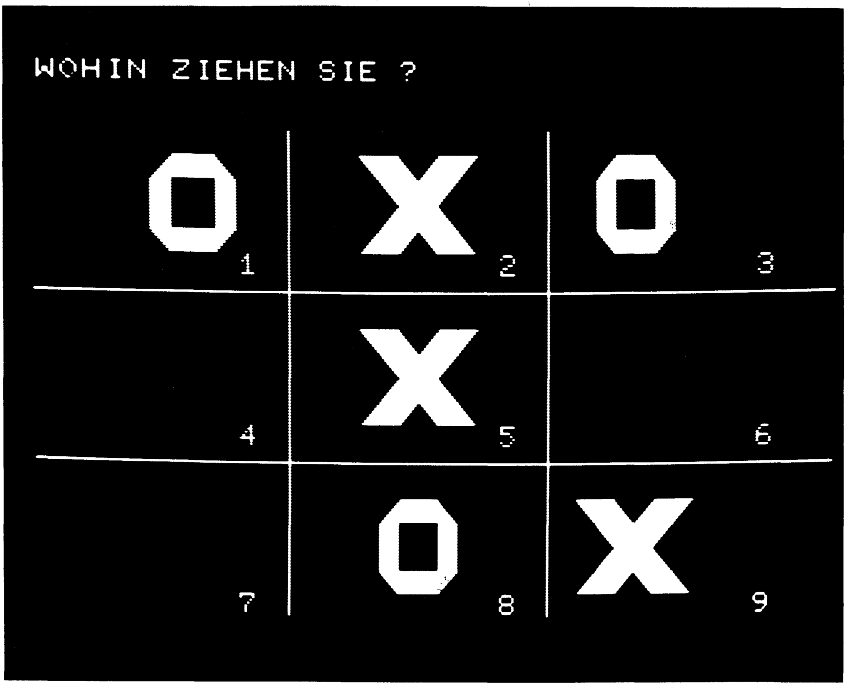
```
100 REM SCHIEBESPIEL
110 :
120 DIMA(16),B(16):TB=8
130 R=216:CR$=CHR$(13)
140 PRINT"□▩▩"TAB(TB)" ┌────┬────┬────┬────┐"
150 FORI=1TO4:FORJ=1TO3:PRINTTAB(TB);"│    │    │    │    │":NEXTJ
160 IFI<4THENPRINTTAB(TB);"├────┼────┼────┼────┤"
170 NEXTI:PRINTTAB(TB);"└────┴────┴────┴────┘"
180 :
190 REM AUFSTELLEN DER ANFANGSPERMUTATION
200 FORI=1TO16:A(I)=I:NEXTI
210 FORI=15TO2STEP-1:J=RND(1)*I+1:T=A(J):A(J)=A(I):A(I)=T:NEXTI
220 FORI=1TO16:B(I)=A(I):IFB(I)=16THEN K=I
230 NEXTI
240 C=0:FORI=1TO16
250 IFA(I)<>ITHENT=A(I):A(I)=A(T):A(T)=T:C=C+1:GOTO250
260 NEXTI
270 IF(CAND1)=0THEN310
280 I=INT(RND(1)*16)+1:IFB(I)=16THEN280
290 J=INT(RND(1)*16)+1:IFB(J)=16ORI=JTHEN280
300 T=B(I):B(I)=B(J):B(J)=T
310 PRINT"▩":PRINT"▩▩▩";
320 FORI=1TO16STEP4:FORJ=0TO3:PRINTTAB(TB+J*5+2);
330 IFB(I+J)<16THENPRINTRIGHT$(STR$(B(I+J)),2);
340 NEXTJ:N=0:PRINT"▩▩▩":NEXTI:POKER,22:PRINT"□"TAB(13)N;".ZUG"
350 :
```

```
360 REM AKTUALISIEREN
370 GETZ$:IFZ$=""THEN370
380 IFZ$<"2"ORZ$>"8"THEN370
390 IFVAL(Z$)AND1THEN370
400 ONVAL(Z$)/2GOTO470,450,430,410
410 T=K+4:IFT>16THEN370
420 GOTO480
430 T=K-1:IF(3ANDT)=0THEN370
440 GOTO480
450 T=K+1:IF(3ANDK)=0THEN370
460 GOTO480
470 T=K-4:IFT<1THEN370
480 B(K)=B(T):B(T)=16
490 I=INT((K-1)/4):J=K-I*4
500 POKER,I*4+5:PRINT"]"TAB(TB+J*5-3)RIGHT$(STR$(B(K)),2)
510 I=INT((T-1)/4):J=T-I*4
520 POKER,I*4+5:PRINT"]"TAB(TB+J*5-3)"   "
530 K=T:N=N+1:POKER,22
540 PRINT"]"TAB(13)STR$(N);" .ZUG";
550 PRINT" ":IF K<16THEN370
560 FORI=1TO16:IFB(I)<>ITHEN370
570 NEXTI:POKER,21:PRINT"]"TAB(13)"GELOEST IM██"
580 PRINT"WOLLEN SIE NOCHEINMAL SPIELEN?";
590 GOSUB630:IFLEFT$(IN$,1)<>"N"THEN140
600 END
610 :
```

```
620 REM UP
630 IN$=" ":ZT=TI:ZC=2:ZD$=CHR$(20)
640 GETZ$:IFZ$<>""THEN670
650 IFZT<=TITHENPRINTMID$(" ▒",ZC,1);"█";:ZC=3-ZC:ZT=TI+15
660 GOTO640
670 Z=ASC(Z$):ZL=LEN(IN$):IF(ZAND127)<32THENPRINT" █";:GOTO710
680 IFFLAND(ZAND127)>64AND(ZAND127)<91THENZ$=CHR$((Z+128)AND255)
690 IFZL>254THEN640
700 IN$=IN$+Z$:PRINTZ$;ZD$;Z$;
710 IFZ=13THENIN$=MID$(IN$,2):PRINTCR$;:RETURN
720 IFZ=20ANDZL>1THENIN$=LEFT$(IN$,ZL-1):PRINT"█";:GOTO640
730 IFZ=141THENZ$=CHR$(-20*(ZL>1)):GOTO 750
740 GOTO 760
750 FORZ=2TOZL:PRINTZ$;:NEXTZ:GOTO630
760 GOTO640
770 PRINT"♥":CLR:GOSUB780:GOTO120
780 R=245:CR$=CHR$(13)
790 IFPEEK(50000)=0THENRETURN
800 R=216
810 RETURN
READY.
```

8. Tic-tac-toe

Tic-tac-toe ist ein bei Schülern sehr beliebtes Spiel, da es sich auf dem kleinsten Zettel spielen läßt: Zwei Spieler markieren abwechselnd auf einem 3 x 3-Quadrat ein "X" bzw. ein "O". Derjenige gewinnt, der eine Zeile, Spalte oder Diagonale mit seinem Zeichen markieren kann.



Wie Nim ist Tic-tac-toe ein sehr altes Spiel, das wahrscheinlich aus Europa kommt. Schon Ovid empfiehlt im 3. Buch seiner "Liebeskunst" den jungen Damen die Beschäftigung mit diesem Spiel, um sich dadurch das Wohlwollen der jungen Männer zu erringen. In England war es so populär, daß der berühmte Dichter Wordsworth darüber ein Gedicht verfaßte. Kein Geringerer als Charles Babbage versuchte mit einer Tic-tac-toe spielenden Maschine so viel Geld zu verdienen, um seine Rechenmaschine (Analytical machine) zu vollenden – es gelang ihm aber nicht.

Tic-tac-toe erfuhr zahlreiche Erweiterungen. Zum einen wurde das Spielfeld auf ein 8 x 8-Quadrat erweitert (vgl. Vier-in-einer-Reihe), zum andren wurde es drei-dimensional ergänzt zum 3 x 3 x 3-Würfel (siehe [1]).

Zum folgenden Programm

Das Programm zeichnet das Tic-tac-toe-Brett und numeriert die Felder, so daß zu einem Zug nur noch die entsprechende Nummer eingegeben werden muß. Der Spieler hat die Wahl, ob er das Zeichen "X" oder "O" nimmt. Der Spieler mit "X" beginnt das Spiel.

```
100 REM TIC-TAC-TOE
110 :
120 DIM F(9):PRINT"[]"
130 FORX=1TO6:PRINT"                    |                    |    ":NEXTX
140 PRINT"]            1  |                2  |               3"
150 PRINT"----------------------+--------------------+--------------------"
160 FORX=1TO6:PRINT"                    |                    |    ":NEXTX
170 PRINT"]            4  |                5  |               6"
180 PRINT"----------------------+--------------------+--------------------"
190 FORX=1TO6:PRINT"                    |                    |    ":NEXTX
200 PRINT"]            7  |                8  |               9"
210 PRINT"#NEHMEN SIE 'X' ODER 'O' ?"
220 GETC$:IFC$=""THEN220
230 IFC$="X"THEN900
240 IFC$<>"O"THEN220
250 P$="O":Q$="X"
260 G=-1:H=1:IFF(5)<>0THEN280
270 F(5)=-1:GOTO800
280 IFF(5)<>1THEN310
290 IFF(1)<>0THEN350
300 F(1)=-1:GOTO800
310 IFF(2)=1ANDF(1)=0THEN750
320 IFF(4)=1ANDF(1)=0THEN750
330 IFF(6)=1ANDF(9)=0THEN790
340 IFF(8)=1ANDF(9)=0THEN790
350 IFG=1THEN370
```

```
360 GOTO420
370 J=3*INT((M-1)/3)+1
380 IF3*INT((M-1)/3)+1=MTHENK=1
390 IF3*INT((M-1)/3)+2=MTHENK=2
400 IF3*INT((M-1)/3)+3=MTHENK=3
410 GOTO430
420 FORJ=1TO7STEP3:FORK=1TO3
430 IFF(J)<>GTHEN470
440 IFF(J+2)<>GTHEN510
450 IFF(J+1)<>0THEN540
460 F(J+1)=-1:GOTO800
470 IFF(J)=HTHEN540
480 IFF(J+2)<>GTHEN540
490 IFF(J+1)<>GTHEN540
500 F(J)=-1:GOTO800
510 IFF(J+2)<>0THEN540
520 IFF(J+1)<>GTHEN540
530 F(J+2)=-1:GOTO800
540 IFF(K)<>GTHEN580
550 IFF(K+6)<>GTHEN620
560 IFF(K+3)<>0THEN650
570 F(K+3)=-1:GOTO800
580 IFF(K)=HTHEN650
590 IFF(K+6)<>GTHEN650
600 IFF(K+3)<>GTHEN650
```

```
610 F(K)=-1:GOTO800
620 IFF(K+6)<>0THEN650
630 IFF(K+3)<>GTHEN650
640 F(K+6)=-1:GOTO800
650 GOTO850
660 IFF(3)=GANDF(7)=0THEN780
670 IFF(9)=GANDF(1)=0THEN750
680 IFF(7)=GANDF(3)=0THEN770
690 IFF(9)=0ANDF(1)=GTHEN790
700 IFG=-1THENG=1:H=-1:GOTO350
710 IFF(9)=1ANDF(3)=0THEN760
720 FORI=2TO9:IFF(I)<>0THEN740
730 F(I)=-1:GOTO800
740 NEXTI
750 F(1)=-1:GOTO800
760 IFF(1)=1THEN720
770 F(3)=-1:GOTO800
780 F(7)=-1:GOTO800
790 F(9)=-1
800 PRINT"█COMPUTER...                        "
810 PRINT"                                    "
820 PRINT"                                    ":PRINT"□□"
830 GOSUB1020
840 GOTO910
850 IFG=1THEN880
860 IFJ=7ANDK=3THEN880
```

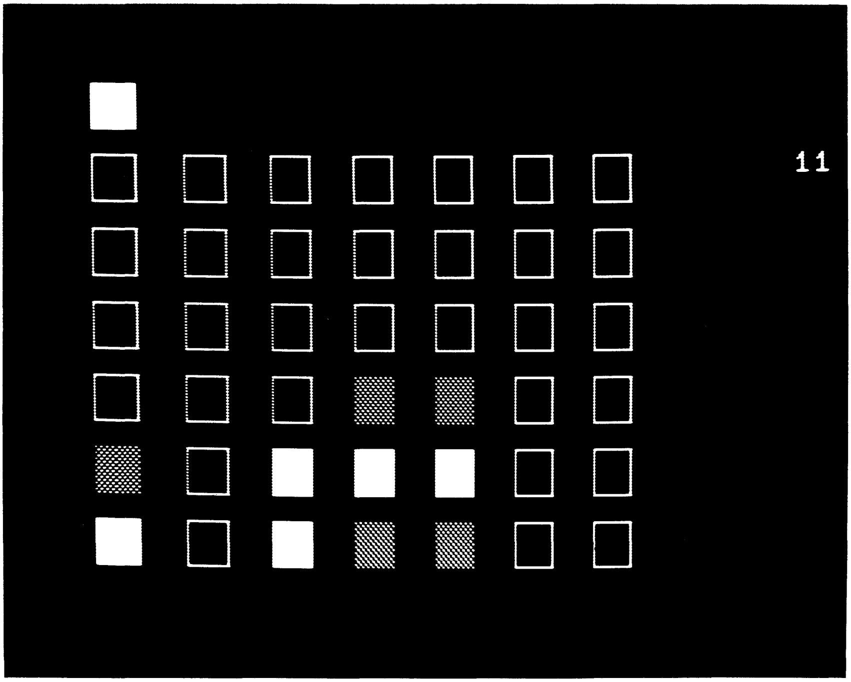
```
870 NEXTK,J
880 IFF(5)=GTHEN660
890 GOTO700
900 P$="X":Q$="O"
910 PRINT:PRINT"▒WOHIN ZIEHEN SIE ?          ███████████"
920 GETC$:IFC$<"0"ORC$>"9"THEN920
930 M=VAL(C$)
940 FORX=1TO2:PRINT"                                                    ":NEXTX:PRINT"⊐⊐"
950 IF M=0 THEN END
960 IFM>9THEN980
970 IFF(M)=0THEN990
980 PRINT"DIESES FELD IST BESETZT":PRINT:PRINT:GOTO910
990 G=1:F(M)=1
1000 GOSUB1020
1010 GOTO260
1020 GOSUB1280
1030 FORI=1TO7STEP3
1040 IFF(I)<>F(I+1)THEN1080
1050 IFF(I)<>F(I+2)THEN1080
1060 IFF(I)=-1THEN1200
1070 IFF(I)=1THEN1190
1080 NEXTI:FORI=1TO3:IFF(I)<>F(I+3)THEN1120
1090 IFF(I)<>F(I+6)THEN1120
1100 IFF(I)=-1THEN1200
```

```
1110 IFF(I)=1THEN1190
1120 NEXTI:FORI=1TO9:IFF(I)=0THEN1140
1130 NEXTI:GOTO1210
1140 IFF(5)<>GTHEN1170
1150 IFF(1)=GANDF(9)=GTHEN1180
1160 IFF(3)=GANDF(7)=GTHEN1180
1170 RETURN
1180 IFG=-1THEN1200
1190 REM
1200 PRINT"■ ■COMPUTER GEWINNT■                              ":GOTO1220
1210 PRINT"■          ■UNENTSCHIEDEN■  ■                     "
1220 PRINT"    NEUES SPIEL ?"
1230 GETC$:IFC$=""THEN1230
1240 IF C$="J" THEN CLR:GOTO 120
1250 IF C$<>"N" THEN 1230
1260 PRINT"■■■■■■","S P I E L E N D E■■■■■■■■"
1270 END
1280 FORI=1TO9
1290 PRINT"■":IFI<4THENAB=2:GOTO1320
1300 IFI>6THENAB=16:GOTO1320
1310 AB=9
1320 FORX=1TOAB+1:PRINT:NEXTX
1330 IFI=1ORI=4ORI=7THENBA=4:GOTO1360
1340 IFI=2ORI=5ORI=8THENBA=15:GOTO1360
1350 BA=26
1360 IFP$="X"THEN1400
```

```
1370 IFF(I)=-1THENGOSUB1430
1380 IFF(I)=1THENGOSUB1490
1390 NEXTI:RETURN
1400 IFF(I)=-1THENGOSUB1490
1410 IFF(I)=1THENGOSUB1430
1420 NEXTI:RETURN
1430 PRINTTAB(BA)"◥▚ ▼ ■◤"
1440 PRINTTAB(BA)" ◥▚  ■◤"
1450 PRINTTAB(BA)" ▚◤  ◥"
1460 PRINTTAB(BA)"▚◤ ■◤◥▚ ◥"
1470 PRINTTAB(BA)"      "
1480 RETURN
1490 PRINTTAB(BA)" ▚◤  ◥"
1500 PRINTTAB(BA)" ▚ ■  ▚ ■"
1510 PRINTTAB(BA)" ▚ ■  ▚ ■"
1520 PRINTTAB(BA)" ◥▚  ■◤"
1530 RETURN
READY.
```

9. Vier-in-einer-Reihe

Wie schon beim Tic-tac-toe erwähnt, kann das Spiel Vier-in-einer-Reihe als Verallgemeinerung aufgefaßt werden. In einem 7 x 6-Quadrat muß ein Spieler zum Gewinn vier Zeichen in einer Zeile, Spalte oder in einer Diagonale setzen. Die Zeichen können jedoch nicht beliebig gesetzt werden, sondern immer nur an die Spitze bereits gesetzter Zeichen.



Zum folgenden Programm

Der Computer zeichnet auf dem Bildschirm das 7 x 6-Quadrat und wartet auf den ersten Zug des Spielers. Dieser wird mittels SHIFT und RETURN in die richtige Spalte gebracht. Dieser Zug wird schraffiert angezeigt, der Gegenzug des Computers reverse.

Da es zahlreiche Kombinationsmöglichkeiten gibt, werden die Computerzüge nicht berechnet, sondern sind durch DATA-Werte vorgegeben. Der Computer wählt also einen dem Spielverlauf entsprechenden String aus.

Dies beschleunigt das Spiel erheblich. Da es keine optimale Strategie gibt, ist der Computer dennoch zu schlagen. Die DATA-Werte stammen aus einem amerikanischen Programm, dessen Autor unbekannt ist.

```
100 REM VIER IN EINER REIHE
110 :
120 PRINT"▢":REM GRAPHIK
130 FORI=0TO24STEP4
140 FORK=0TO600STEP120:J=32929+I+K
150 POKEJ,79:POKEJ+1,80:POKEJ+40,76
160 POKEJ+41,122:NEXTK:NEXTI
170 DIMX$(42),Y$(69),A(7),B(69)
180 :
190 E=32809:F=32929:G=32963
200 FORI=1TO42:READX$(I):NEXTI:RESTORE
210 PRINT"▪";TAB(7)"▪VIER IN EINER REIHE"
220 PRINT"DER STEIN WIRD DURCH DIE SPACE-"
230 PRINT"TASTE IN POSITION GEBRACHT."
240 PRINT"ER FAELLT NACH DRUECKEN VON RETURN"
250 FORI=1TO42:B=1
260 Z=VAL(MID$(X$(I),B,2))
270 IFZ=0THEN320
280 IFLEN(Y$(Z))=6THENY$(Z)=Y$(Z)+RIGHT$(STR$(I),2):GOTO300
290 Y$(Z)=RIGHT$(STR$(I),2)+Y$(Z)
300 B=B+2:GOTO260
310 NEXTI
320 FORI=32775TO32927:POKEI,32:NEXTI
330 PRINT"▪";TAB(10)"SPIELSTAERKE(1-2)?"
340 GETA$:IFA$<>""THEN340
350 GETSS:IFSS<>1ANDSS<>2THEN350
```

```
360 FORI=32775TO32800:POKEI,32:NEXTI
370 A=E:B=1:Z=Z+1
380 POKEG,ASC(RIGHT$(STR$(Z),2))
390 POKEG+1,ASC(RIGHT$(STR$(Z),1))
400 IFZ>41THENPRINT"UNENTSCHIEDEN":POKEG+1,50:GOTO650
410 GOSUB1350:POKEA,B%:POKEA+1,B%
420 POKEA+40,B%:POKEA+41,B%:D=5
430 GETA$:IFA$<>""THEN430
440 GETA$:IFA$=" "THENGOSUB1370
450 IFA$="E"THEN1220
460 IFA$<>CHR$(13)THEN440
470 IFPEEK(A+120)>79THEN440
480 POKEA,32:POKEA+1,32:POKEA+40,32:POKEA+41,32
490 A=A+120:POKEA,B%:POKEA+1,B%:POKEA+40,B%
500 POKEA+41,B%:IFPEEK(A+120)<>79THEN530
510 POKEA,79:D=D-1
520 POKEA+1,80:POKEA+40,76:POKEA+41,122:GOTO490
530 A=7*D+B:B=1:R=1:IFB%=102THENR=-1
540 D=VAL(MID$(X$(A),B,2))
550 IFD=0THEN700
560 B(D)=B(D)+R:IFB(D)↑2=16THEN580
570 B=B+2:GOTO540
580 B$="SPIELER":IFB%=102THENB$="COMPUTER"
590 PRINT"";B$;" GEWINNT"
600 FORR=1TO7STEP2
610 B=VAL(MID$(Y$(D),R,2))/7
```

```
620 T=120*(6-INT(B-.1)):S=(B-INT(B-.1))*28-4
630 POKEE+S+T,79:POKEE+S+T+41,122
640 NEXTR
650 PRINT"■NEUES SPIEL (J/N) ?:"
660 GETA$:IFA$<>""THEN660
670 GETA$:IFA$="J"THEN1220
680 IFA$="N"THEN END
690 GOTO670
700 IFB%=102THEN370
710 POKEG,ASC(RIGHT$(STR$(Z),2))
720 POKEG+1,ASC(RIGHT$(STR$(Z),1))
730 POKEE,102:POKE32810,102
740 POKE32849,102:POKE32850,102
750 IFZ<2THENS=12:GOTO1280
760 FORI=1TO69:IFB(I)>-3THEN820
770 FORK=1TO8STEP2:A=VAL(MID$(Y$(I),K,2))/7
780 T=120*(6-INT(A-.1)):S=(A-INT(A-.1))*28-4
790 S=INT(S+.5)
800 IFPEEK(E+T+S)<>79THENNEXTK
810 IFPEEK(F+T+S)<>79THEN1280
820 NEXTI:FORI=1TO69:IFB(I)<3THEN880
830 FORK=1TO8STEP2:A=VAL(MID$(Y$(I),K,2))/7
840 T=120*(6-INT(A-.1)):S=(A-INT(A-.1))*28-4
850 S=INT(S+.5)
860 IFPEEK(E+T+S)<>79THENNEXTK
870 IFPEEK(F+T+S)<>79THEN1280
```

```
880 NEXTI:FORM=SSTO2:FORI=69TO1STEP-1:IFB(I)<>2THEN980
890 FORK=1TO10STEP2:A=VAL(MID$(Y$(I),K,2)):IFA=0THEN980
900 T=120*(6-INT(A/7-.1)):S=(A-INT(A/7-.1)*7)*4-4
910 IFPEEK(E+T+S)<>79THENNEXTK
920 IFPEEK(F+T+S)=79THENNEXTK
930 A=(7-T/120)*7+S/4+1:J=1:IFA>42THEN1280
940 L=VAL(MID$(X$(A),J,2))
950 IFL=0THEN1280
960 IFB(L)>MORB(L)<-MTHENNEXTK:GOTO980
970 J=J+2:GOTO940
980 NEXTI:NEXTM
990 FORM=SSTO2:FORI=1TO13
1000 S=INT(RND(1)*6)*4
1010 IFPEEK(F+S)>79THEN1100
1020 FORT=0TO600STEP120
1030 IFPEEK(F+T+S)=79THENNEXTT
1040 IFT=120THEN1280
1050 A=(6-(T-120)/120)*7+S/4+1:J=1
1060 K=VAL(MID$(X$(A),J,2))
1070 IFK=0THEN1280
1080 IFB(K)>MORB(K)<-MTHEN1100
1090 J=J+2:GOTO1060
1100 NEXTI:NEXTM
1110 FORS=0TO26STEP4
1120 IFPEEK(F+S)>79THEN1210
1130 FORT=0TO600STEP120
```

```
1140 IFPEEK(F+T+S)=79THENNEXTT
1150 IFT=120THEN1280
1160 A=(6-(T-120)/120)*7+S/4+1:J=1
1170 K=VAL(MID$(X$(A),J,2))
1180 IFK=0THEN1280
1190 IFB(K)=3THEN1210
1200 J=J+2:GOTO1170
1210 NEXTS:PRINT"■■COMPUTER GIBT AUF.":GOTO650
1220 PRINT"▯":FORA=0TO24STEP4
1230 FORB=0TO600STEP120:Z=F+A+B
1240 POKEZ,79:POKEZ+1,80:POKEZ+40,76
1250 POKEZ+41,122:NEXTB:NEXTA
1260 FORI=1TO69:B(I)=0:NEXTI
1270 Z=0:B%=102:GOTO330
1280 A=E:B=1:Z=Z+1:D=5
1290 POKEG,ASC(RIGHT$(STR$(Z),2))
1300 POKEG+1,ASC(RIGHT$(STR$(Z),1))
1310 R=0:GOSUB1350:IFS=0THEN480
1320 R=R+1:GOSUB1370
1330 IFR=>S/4THEN480
1340 GOTO1320
1350 IFB%=160THENB%=102:RETURN
1360 B%=160:RETURN
1370 B=B+1:POKEA,32:POKEA+1,32
1380 POKEA+40,32:POKEA+41,32:A=A+4
1390 IFB>7THENB=1:A=E:GOTO1390
```

```
1400 POKEA,B%:POKEA+1,B%:POKEA+40,B%
1410 POKEA+41,B%:RETURN
1420 REM COMPUTERSTRATEGIE
1430 DATA"01254900","0102285200"
1440 DATA"010203315500","01020304345575800"
1450 DATA"020304375900","0304406100"
1460 DATA"04436400","0525264700"
1470 DATA"05062829495000"
1480 DATA"050607313252535800"
1490 DATA"0506070834355556596000"
1500 DATA"060708373861625700"
1510 DATA"070840416465000","0843446700"
1520 DATA"09252627460 0"
1530 DATA"091028293047485800"
1540 DATA"09101113132334950515960000"
1550 DATA"0910111234353652535461626300"
1560 DATA"10111237383964656655560 0"
1570 DATA"111240414267685700"
1580 DATA"124344456900","1325262758 00"
1590 DATA"131428293059604600"
1600 DATA"1314153132336162634748 00"
1610 DATA"13141516343536646566495051 00"
1620 DATA"1415163738396768525354 00"
1630 DATA"151640414269555600"
1640 DATA"164344455700","2627176000"
1650 DATA"171829306263 00"
```

```
1660 DATA"1718193233365664600"
1670 DATA"171819203536474868670O"
1680 DATA"1819203839505169OO"
1690 DATA"192041425354OO","2044455600"
1700 DATA"27216300","2122306600"
1710 DATA"212223336800","2122232436694600"
1720 DATA"222324394800","2324425100"
1730 DATA"24455400"
READY.
```

10. Solitaire

Dieses Spiel soll von einem adligen Gefangenen in der Bastille im 18. Jahrhundert erfunden worden sein und erfreute sich in Frankreich großer Beliebtheit. Aber auch der große deutsche Mathematiker Leibniz hat sich schon 1716 ausführlich damit beschäftigt.

Eine umfassende mathematische Analyse, "The Game of Solitaire" schrieb E. Bergholt 1920. Er numerierte die 33 Löcher des Spielbretts wie folgt:

		o 37	o 47	o 57		
		o 36	o 46	o 56		
o 15	o 25	o 35	o 45	o 55	o 65	o 75
o 14	o 24	o 34	o 44	o 54	o 64	o 74
o 13	o 23	o 33	o 43	o 53	o 63	o 73
		o 32	o 42	o 52		
		o 31	o 41	o 51		

Alle Löcher des Spielbretts, mit Ausnahme des mittleren, Loch 44, sind mit einem Stecker besetzt. Ziel des Spieles ist es, durch Überspringen, ähnlich wie beim Damespiel, möglichst viele der 32 Stecker aus dem Spiel zu nehmen, wobei vor– und zurück, aber nicht diagonal gesprungen werden darf.

Idealziel ist es, 31 Stecker so abzuräumen, daß der letzte verbleibende im Loch 44 steckt. Das setzt aber schon eine erhebliche Meisterschaft in diesem Spiel voraus.

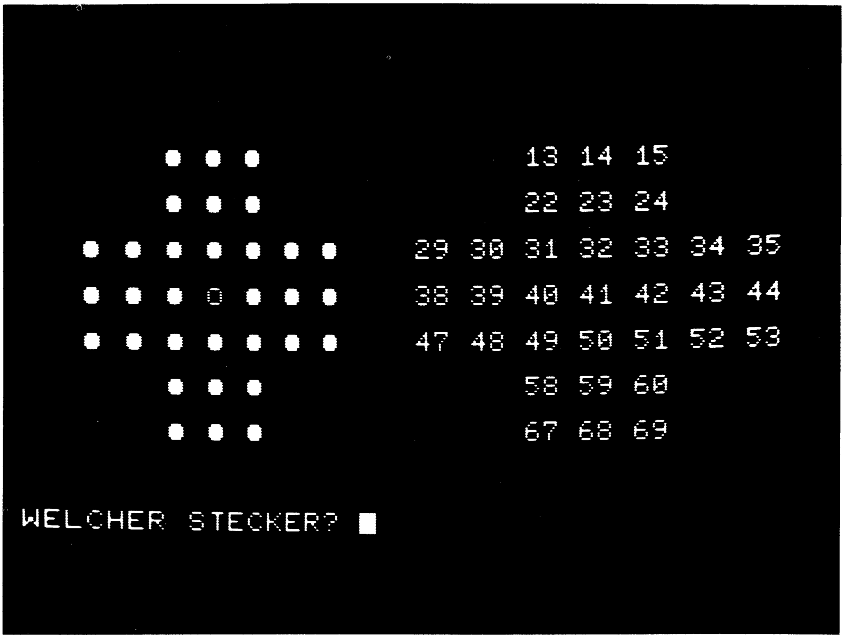
Für alle Spielernaturen sei verraten, daß es eine optimale Lösung gibt (vergl. [5]):

46–44,65–45,57–55,54–56,52–54,73–53,43–63,75–73–53,35–55,
15–35,23–43–63–65–45–25,37–57–55–53,31–33,34–32,51–31–33,
13–15–35,36–34–32–52–54–34,24–44.

Diese Spielweise, die ebenfalls von Bergholt (1912) stammt, umfaßt nur 18 Sprünge, wobei allerdings die Mehrfachsprünge nur einmal gezählt werden.

Zum folgenden Programm

Im Programm wird das Solitaire-Brett anders und zwar diagonal durchnumeriert; dies vereinfacht die Kontrolle der eingegebenen Sprünge. Damit sich der Spieler diese Numerierung nicht merken muß, wird sie stets am Bildschirm miteingeblendet.



Jeder Sprung wird durch Angabe des Start- und Ziellochs eingegeben. Das Spiel endet, wenn der Spieler keine Zugmöglichkeit mehr besitzt.

```
100 REM SOLITAIRE
110 :
120 DIM W(70),F(9,9)
130 PRINT"▯"
140 PRINTTAB(3)"▪SPIELBRETT MIT NUMERIERUNG▪"
150 PRINT"         O   O   O"
160 PRINT"        13  14  15":PRINT
170 PRINT"         O   O   O"
180 PRINT"        22  23  24":PRINT
190 PRINT" O   O   O   O   O   O   O"
200 PRINT"29  30  31  32  33  34  35":PRINT
210 PRINT" O   O   O   O   O   O   O"
220 PRINT"38  39  40  41  42  43  44":PRINT
230 PRINT" O   O   O   O   O   O   O"
240 PRINT"47  48  49  50  51  52  53":PRINT
250 PRINT"         O   O   O"
260 PRINT"        58  59  60":PRINT
270 PRINT"         O   O   O"
280 PRINT"        67  68  69":PRINT
290 FOR I=1 TO 9
300 FOR J=1 TO 9
310 IF I=4 OR I=5 OR I=6 THEN 350
320 IF J=4 OR J=5 OR J=6 THEN 350
330 F(I,J)=-5
340 GOTO 370
350 IF I=1 OR J=1 OR I=9 OR J=9 THEN 330
```

```
360 F(I,J)=5
370 NEXT J
380 NEXT I
390 F(5,5)=0:GOSUB 570
400 FOR W=1 TO 33
410 READ M
420 W(M)=-7:NEXT W
430 W(41)=-3
440 INPUT"WELCHER STECKER";Z
450 IF W(Z)=-7 THEN 470
460 PRINT"UNERLAUBTER ZUG,NOCHEINMAL!":GOTO 440
470 INPUT"WOHIN";X
480 IF W(X)=0 OR W(X)=-7 THEN 460
490 IF Z=X THEN 440
500 IF(Z+X)/2=INT((Z+X)/2) THEN 520
510 GOTO 440
520 IF(ABS(Z-X)-2)*(ABS(Z-X)-18)<>0 THEN 460
530 GOSUB 810
540 GOSUB 570
550 GOSUB 1070
560 GOTO 440
570 REM AUSGABE
580 FOR K=1 TO 9
590 FOR L=1 TO 9
600 IF K=1 OR K=9 OR L=1 OR L=9 THEN 630
610 IF K=4 OR K=5 OR K=6 THEN 640
```

```
620 IF L=4 OR L=5 OR L=6 THEN 640
630 GOTO 680
640 IF F(K,L)<>5 THEN 670
650 PRINTTAB(2*L-2);" ●";
660 GOTO 680
670 PRINTTAB(2*L-2);" o";
680 NEXT L
690 PRINT:PRINT
700 NEXT K
710 PRINT"IIIIIIIIIIIIIIIIII"
720 PRINTTAB(20)"       13 14 15":PRINT
730 PRINTTAB(20)"       22 23 24":PRINT
740 PRINTTAB(20)"29 30 31 32 33 34 35"
750 PRINTTAB(20)"38 39 40 41 42 43 44"
760 PRINTTAB(20)"47 48 49 50 51 52 53"
770 PRINTTAB(20)"       58 59 60":PRINT
780 PRINTTAB(20)"       67 68 69"
790 PRINT"██"
800 RETURN
810 REM AKTUALISIEREN DES SPIELBRETTS
820 J=1:FOR K=1 TO 9
830 FOR L=1 TO 9
840 IF J<>Z THEN 1030
850 IF J+2<>X THEN 900
860 IF F(K,L+1)=0 THEN 460
870 F(K,L+2)=5
```

```
880 F(K,L+1)=0:W(J+1)=-3
890 GOTO 1010
900 IF J+18<>X THEN 940
910 IF F(K+1,L)=0 THEN 460
920 F(K+2,L)=5:F(K+1,L)=0:W(J+9)=-3
930 GOTO 1010
940 IF J-2<>X THEN 980
950 IF F(K,L-1)=0 THEN 460
960 F(K,L-2)=5:F(K,L-1)=0:W(J-1)=-3
970 GOTO 1010
980 IF J-18<>X THEN 1030
990 IF F(K-1,L)=0 THEN 460
1000 F(K-2,L)=5:F(K-1,L)=0:W(J-9)=-3
1010 W(Z)=-3:W(X)=-7
1020 F(K,L)=0:GOTO 1060
1030 J=J+1
1040 NEXT L
1050 NEXT K
1060 RETURN
1070 REM PRUEFEN AUF ZUGMOEGLICHKEIT
1080 F=0
1090 FOR I=2 TO 8
1100 FOR J=2 TO 8
1110 IF F(I,J)<>5 THEN 1290
1120 F=F+1
1130 FOR A=I-1 TO I+1
```

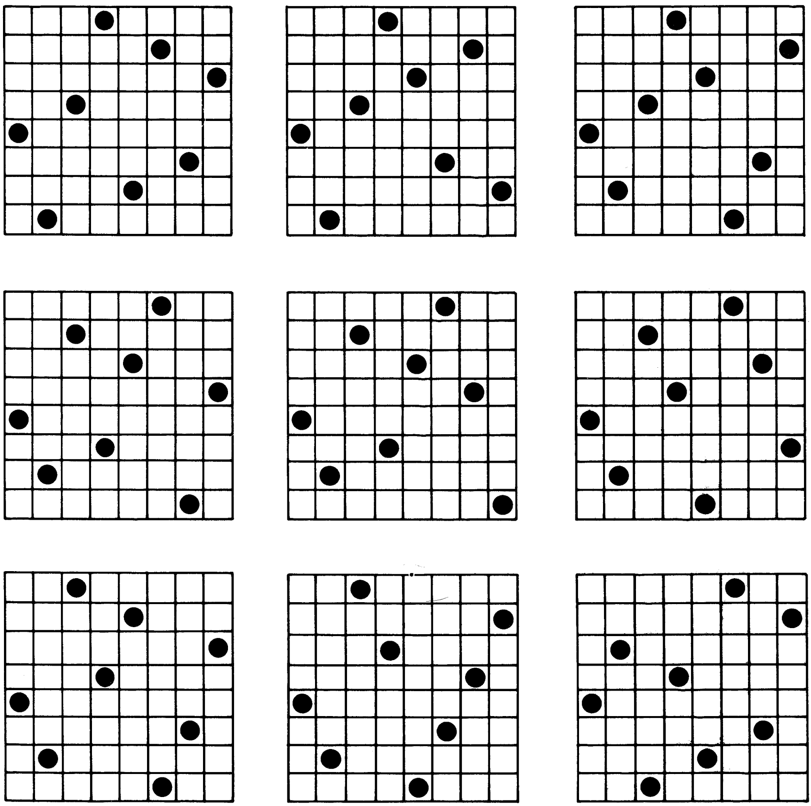
```
1140 T=0
1150 FOR B=J-1 TO J+1
1160 T=T+F(A,B)
1170 NEXT B
1180 IF T<>10 THEN 1200
1190 IF F(A,J)<>0 THEN RETURN
1200 NEXT A
1210 FOR K=J-1 TO J+1
1220 T=0
1230 FOR L=I-1 TO I+1
1240 T=T+F(L,K)
1250 NEXT L
1260 IF T<>10 THEN 1280
1270 IF F(I,K)<>0 THEN RETURN
1280 NEXT K
1290 NEXT J
1300 NEXT I
1310 PRINT"SPIEL ZU ENDE"
1320 PRINT"ES BLEIBEN";F;"STECKER UEBRIG"
1330 IF F=1 THEN PRINT"GRATULIERE,PERFEKTES SPIEL"
1340 END
1350 DATA 13,14,15,22,23,24,29,30,31,32,33,34,35,38,39,40,41
1360 DATA 42,43,44,47,48,49,50,51,52,53,58,59,60,67,68,69
READY.
```

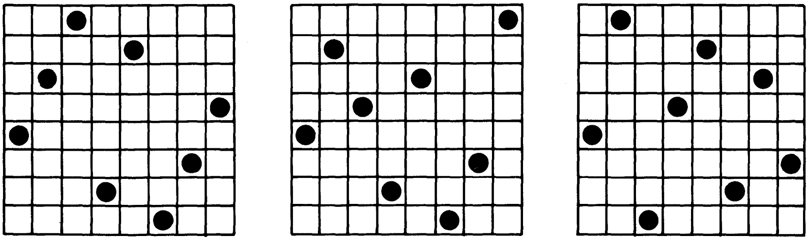
11. Acht-Damen-Problem

Ein inzwischen klassisches Problem ist das sog. Acht-Damen-Problem. Es handelt sich darum, die Position von 8 Damen auf einem Schachbrett so zu bestimmen, daß sie sich nach den üblichen Schachregeln nicht bedrohen.

Dieses Problem wurde 1848 von Max Bezzel erdacht und in einer Berliner Schachzeitung veröffentlicht. Auch der berühmteste deutsche Mathematiker Gauß beschäftigte sich damit, konnte jedoch nicht alle Lösungen finden. Es gibt nämlich nicht weniger als 92 Lösungen, von denen jedoch nur 12 nicht durch Spiegelung oder Drehung aus den anderen hervorgehen.

Diese 12 Lösungen sind (vgl. [5]):

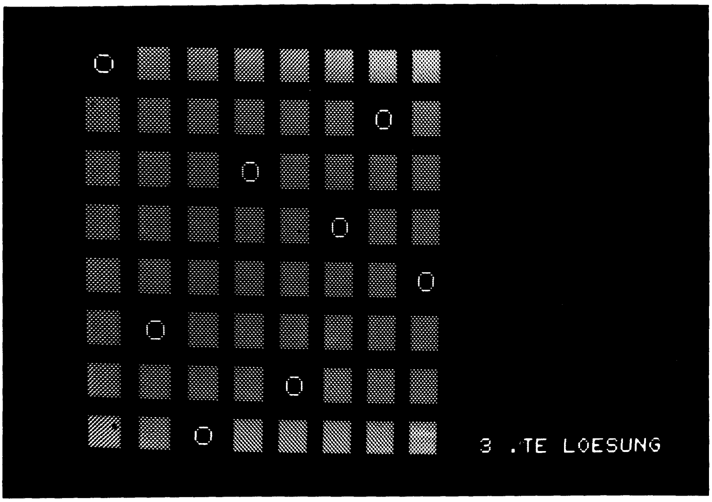




Sie wurden 1850 von Frank Nauk in der Leipziger Illustrierten publiziert. Der Nachweis, daß diese 12 Lösungen wirklich vollständig sind, gelang erst 1874 dem Engländer Glaisher.

Zum folgenden Programm

Da offensichtlich in jeder Zeile nur eine Dame stehen kann, kann der Zeilenindex z mit der Anzahl der Damen gleichgesetzt werden. Durch Weiterzählen von z wird also eine neue Dame aufs Brett gebracht, durch Herunterzählen vom Brett entfernt. Das Feld D(z) gibt jeweils die Spaltenzahl an. Die Bedrohung der Damen kann somit durch D(s)=D(z) oder |D(z)−D(s)|=z−s geprüft werden. Durch probeweises Setzen von Damen wird versucht eine Lösungsstellung zu erreichen, ist dies nicht möglich, werden die Damen nach dem oben angegebenen Verfahren wieder vom Brett genommen.



Da die Berechnung der Stellungen sehr aufwendig ist, benötigt der Rechner ca. 15 Minuten zur vollständigen Lösung.

```
100 REM 8-DAMEN-PROBLEM
110 :
120 DIM D(8)
130 Z=0
140 C=0
150 PRINT"■■■■■■■■■■■■■■■■■■■■■■BITTE WARTEN"
160 Z=Z+1
170 D(Z)=1
180 IF Z=1 THEN 220
190 FOR S=1 TO Z-1
200 IF D(Z)=D(S) OR ABS(D(Z)-D(S))=Z-S THEN 360
210 NEXT S
220 IF Z<8 THEN 160
230 REM AUSGABE
240 PRINT"■"
250 C=C+1
260 FOR S=1 TO 8
270 FOR L=1 TO 8
280 IF D(S)=L THEN PRINT"■■■■■■ ";:GOTO 300
290 PRINT"■■■■■■ ";
300 NEXT L
310 PRINT"■■"
320 NEXT S
330 PRINT"■■";TAB(25)C;".TE LOESUNG"
340 REM DAME VOM FELD
350 Z=Z-1
```

```
 360 D(Z)=D(Z)+1
 370 IF D(Z)<=8 THEN 180
 380 Z=Z-1
 390 IF Z>0 THEN 360
 400 END
READY.
```

12. Springerzug

Neben dem 8-Damen-Problem gibt es noch zahllose andere Aufgaben, die sich mit Eigenschaften eines Schachbretts beschäftigen.

Ein solches Problem verlangt, die Bahn eines Springers auf einem Schachbrett zu bestimmen, der sich nach den üblichen Regeln bewegt und jedes Feld genau einmal berühren soll.

Eine mögliche Lösung ist

1	42	13	26	3	60	15	28
24	37	2	41	14	27	4	61
43	12	25	38	59	62	29	16
36	23	40	63	48	51	56	5
11	44	49	52	39	58	17	30
22	35	64	47	50	55	6	57
45	10	33	20	53	8	31	18
34	21	46	9	32	19	54	7

dabei geben die Nummern die Zugfolge des Springers an.

Interessant ist, daß es auch ein magisches Quadrat gibt, das einen vollständigen Springerzug beschreibt:

46	55	44	19	58	9	22	7
43	18	47	56	21	6	59	10
54	45	20	41	12	57	8	23
17	42	53	48	5	24	11	60
52	3	32	13	40	61	34	25
31	16	49	4	33	28	37	62
2	51	14	29	64	39	26	35
15	30	1	50	27	36	63	38

Zum folgenden Programm

Zur Lösung von Problemen, wie die des Springerzugs, wendet man sog. Backtracking-Verfahren an. Dies sind Verfahren, die auf Probieren beruhen und, falls sie in eine Sackgasse geraten, wieder an einem Punkt starten, der eine Weiterentwicklung erlaubt. Eine solche Methode wurde auch beim 8-Damen-Problem benützt. Ein Backtracking-Verfahren für ein Labyrinth findet sich z.B. in [2]. Diese Methoden sind im allgemeinen rekursiv, d.h. sie greifen auf sich selbst zurück. Im allgemeinen kann man das in BASIC nur schwer realisieren. Deswegen wird beim folgenden Programm ein zufälliger Weg gewählt, der natürlich nicht allzu oft zu einem vollständigen Springerweg auf dem Schachbrett führt.

```
100 REM SPRINGER-ZUG
110 :
120 REM INITIALISIEREN
130 DIMB(8,8)
140 :
150 FOR I=1 TO 8
160 FOR J=1 TO 8
170 F(I,J)=0
180 NEXT J
190 NEXT I
200 Z=2
210 :
220 REM EINLESEN DES STARTPUNKTS
230 READ I,J
240 F(I,J)=-1
250 :
260 REM ZUFALLSSPRUENGE
270 S=9
280 C=0
290 FOR K=I-2 TO I+2
300 IF K=I THEN 430
310 IF ABS(K-4.5)>4 THEN 430
320 D=3-ABS(K-I)
330 FOR L=J-D TO J+D STEP 2*D
340 IF ABS(L-4.5)>4 THEN 420
350 IF F(K,L)<0 THEN 420
```

```
360 C=C+1
370 IF C<>S THEN 420
380 I=K:J=L
390 F(I,J)=-Z
400 Z=Z+1
410 GOTO 270
420 NEXT L
430 NEXT K
440 IF C=0 THEN 470
450 S=INT(C*RND(1)+1):GOTO 280
460 :
470 PRINT"SPRINGERZUG"
480 FOR I=1 TO 8
490 FOR J=1 TO 8
500 PRINTTAB(4*J)ABS(F(I,J));
510 NEXT J
520 PRINT:PRINT
530 NEXT I
540 :
550 PRINT"";Z-1;"SPRUENGE"
560 PRINT"NEUER DURCHGANG (J/N)?"
570 INPUT A$
580 IF MID$(A$,1,1)="J" THEN RESTORE:GOTO 200
590 END
600 :
610 DATA 1,1
READY.
```

13. Münzwechsel-Problem

Eine bekannte Aufgabe ist das Münzwechsel-Problem. Es beantwortet Fragen wie: Auf wieviele Arten kann man eine DM in Pfennige, 2-, 5-, 10- und 50-Pfennigstücke wechseln?

Auf wieviele Arten kann man einen Brief, der mit 1 DM frankiert werden soll, mit 10-, 20- und 30-Pfennig-Marken bekleben?

Zum folgenden Programm

Zur Lösung wird ein rekursives Verfahren verwendet; d.h. die Wechselmöglichkeiten für größere Münzen werden aufbauend aus denen für kleinere Münzen berechnet.

Beispiel Bundesrepublik: Gesucht ist die Anzahl der Wechselmöglichkeiten für 1 DM. Da es 5 kleinere Münzen gibt, müssen in Zeile 570 folgende DATA-Werte erscheinen: 5, 1, 2, 5, 10, 50. Die letzten 5 Werte sind die Pfennigbeträge der Münzen. Das Programm liefert den Wert 2498. Es gibt also 2498 Möglichkeiten eine DM zu wechseln.
Zu beachten ist, daß die Münzen als nicht unterscheidbar angesehen werden. Es gibt also nur eine Möglichkeit 1 DM in Pfennige zu wechseln, nämlich einmal in 100 einzelne Pfennige.

Beispiel Schweiz: In der Schweiz gibt es 1, 2, 5, 10, 20 und 50 Rappen-Stücke. Eingabe der DATA-Werte 6, 1, 2, 5, 10, 20, 50 in Zeile 570 liefert als Ergebnis 4562 Wechselmöglichkeiten für einen Schweizer Franken.

Beispiel USA: In den Vereinigten Staaten existieren Münzen zu 1, 5, 10, 25 und 50 Cents. Es müssen somit die DATA-Werte 5, 1, 5, 10, 25, 50 in Zeile 570 eingegeben werden. Das Ergebnis ist: 1 Dollar kann auf 292 Arten gewechselt werden.

```
100 REM MUENZWECHSELPROBLEM
110 :
120 DIM B(20),W(100,20)
130 REM EINLESEN DER WERTE
140 READ D : REM SCHRITTWEITE FUER AUSZUGEBENDE MUENZBETRAEGE
150 READ M :REM GROESSTE MUENZE
160 READ Z : REM ANZAHL DER MUENZEN
170 FOR J=1 TO Z
180 READ B(J):REM MUENZBETRAEGE
190 NEXT J
200 :
210 REM WECHSELMOEGLICHKEITEN FUER PFENNIGE
220 N=INT(M/D)
230 FOR I=0 TO N
240 W(I,1)=1
250 NEXT I
260 :
270 REM ITERATIV-REKURSIVE BERECHNUNG DER WECHSELMOEGLICHKEITEN
280 FOR J=2 TO Z
290 B=B(J)
300 FOR I=0 TO N
310 T=I*D
320 S=0
330 FOR K=0 TO T STEP B
340 R=T-K
350 S=S+W(R/D,J-1)
```

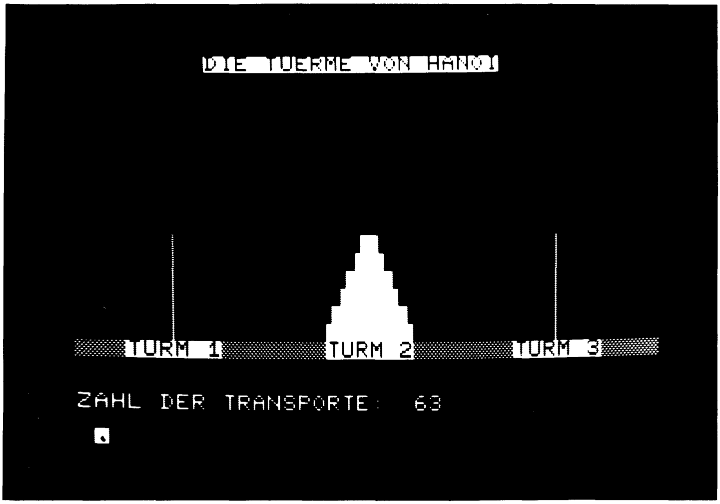
```
360 NEXT K
370 W(I,J)=S
380 NEXT I
390 NEXT J
400 :
410 PRINT"MUENZWECHSEL-PROBLEM"
420 PRINT"BETRAG WECHSELMOEGL.F.FOLG.MUENZSORTE"
430 FOR J=1 TO Z
440 PRINTTAB(6*J)B(J);
450 NEXT J
460 PRINT
470 FOR I=0 TO N
480 PRINT I*D;
490 FOR J=1 TO Z
500 PRINTTAB(6*J)W(I,J);
510 NEXT J
520 PRINT
530 NEXT I
540 END
550 :
560 DATA 5,100
570 DATA 5,1,2,5,10,50
READY.
```

14. Die Türme von Hanoi

In der indischen Stadt Benares stehen 3 Säulen im Tempel des Brahma. Auf einer dieser Säulen sind 64 goldene Scheiben verschiedenen Durchmessers aufgetürmt. Die Welt wird in Schutt und Asche fallen, wenn die Mönche des Tempels die Scheiben einer Säule auf eine andere unter folgenden Bedingungen gelegt haben:

- es darf nie mehr als eine Scheibe gleichzeitig bewegt werden
- es darf nie eine größere Scheibe auf einer kleineren liegen.

Mit dieser mystischen Sage warb der französische Mathematiker Edouard Lucas, als er 1883 unter einem Pseudonym sein Spiel "Türme von Hanoi" auf den Markt brachte.



Man kann zeigen, daß man zum Umlegen von n Scheiben gemäß den oben erwähnten Bedingungen mindestens

$$2^n - 1$$

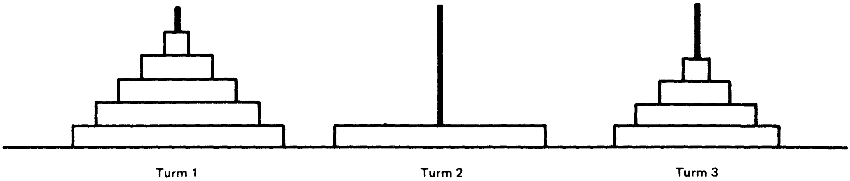
Scheibenbewegungen machen muß. Berechnet man für jede Bewegung 1 Sekunde, so benötigen die Mönche für 64 Scheiben

$$2^{64} - 1 = 18.446.744.073.709.551.615 \text{ Sekunden.}$$

Die anfangs erwähnte Sage entspricht also der Hoffnung, daß unser Planet noch die nächsten 500 Milliarden Jahre überstehen wird, was durchaus optimistisch sein dürfte.

Zum folgenden Programm

Das Programm führt die Scheibenbewegungen graphisch durch.
Der Algorithmus wird rekursiv geführt:



Wie die Zeichnung zeigt, kann das Problem von n Scheiben am Turm 1 auf das Problem für n–1 Scheiben am Turm 3 zurückgeführt werden. Dieses Rückwärts-Rechnen und Vertauschen der Türme muß in BASIC explizit programmiert werden (hier in Form eines Unterprogramms); in anderen Programmiersprachen wie PASCAL kann das Verfahren direkt rekursiv definiert werden.

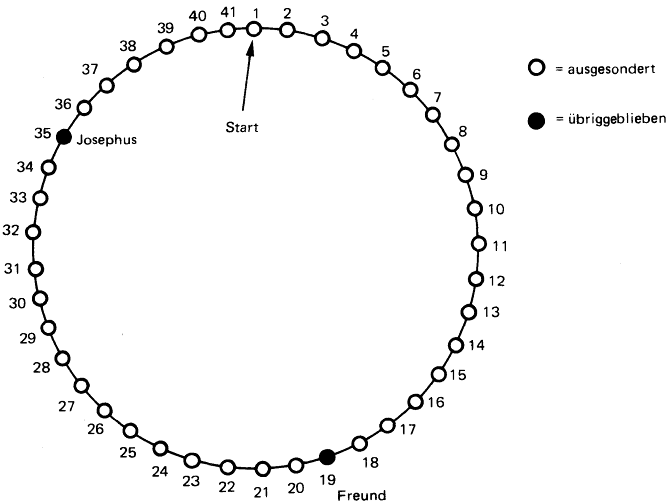
```
100 REM TUERME VON HANOI
110 :
120 PRINT"DIE TUERME VON HANOI":PRINT
130 INPUT"ANZAHL DER SCHEIBEN <=12";N%
140 DIM F%(50),T%(2,N%),S$(N%)
150 K%=0:N1%=N%:T1%=1:T2%=2:T3%=3
160 T$="TURM 1TURM 2TURM 3"
170 GOSUB 220
180 GOSUB 310
190 PRINT"":PRINT"ZAHL DER TRANSPORTE: ";2↑N1%-1
200 END
210 :
220 PRINT"DIE TUERME VON HANOI"
230 RESTORE: FOR I=0 TO N%: READ S$(I): T%(0,I)=N%-I+1: NEXT I
240 FOR I=1 TO 15-N%: PRINT: NEXT I
250 FOR I=N% TO 1 STEP -1: FOR J=0 TO 2
260 PRINT S$(T%(J,I));:NEXT J:PRINT:NEXT I
270 PRINTT$
280 T%(0,0)=N%
290 RETURN
300 :
310 F%(K%)=N%:F%(K%+1)=T1%:F%(K%+2)=T2%:F%(K%+3)=T3%
320 IF N%>1 THEN N%=N%-1:K%=K%+4:J%=T2%:T2%=T3%:T3%=J%:GOSUB 310
330 GOSUB 380
340 IF N%>1 THEN N%=N%-1:K%=K%+4:J%=T1%:T1%=T3%:T3%=J%:GOSUB 310
350 IF K%>0 THEN K%=K%-4:N%=F%(K%):T1%=F%(K%+1):T2%=F%(K%+2):T3%=F%(K%+3)
```

```
360 RETURN
370 :
380 T4%=T1%-1:T5%=T2%-1:L%=T%(T4%,0)
390 PRINT"■":FOR I=1 TO 15-L%:PRINT"■";:NEXT I
400 J%=T%(T4%,L%):T%(T4%,L%)=0:T%(T4%,0)=L%-1
410 FOR I=0 TO 2:PRINT S$(T%(I,L%));:NEXT I
420 FOR I=1 TO L%:PRINT"■";:NEXT I
430 M%=T%(T5%,0)+1:T%(T5%,0)=M%:T%(T5%,M%)=J%
440 PRINT"■":FOR I=1 TO 15-M%:PRINT"■";:NEXT I
450 FOR I=0 TO 2:PRINT S$(T%(I,M%));:NEXT I
460 FOR I=1 TO M%+1:PRINT"■";:NEXT I
470 RETURN
480 :
490 DATA"     |     ","     ■ ■     ","     ■ ■     "
500 DATA"   ■   ■   ","   ■   ■   ","   ■     ■   "
510 DATA"  ■    ■  ","  ■    ■  ","  ■     ■  "
520 DATA" ■     ■  "," ■     ■  ","■       ■ "
530 DATA"■        ■ "
READY.
```

15. Josephus-Problem

In einem der ältesten Rechenbücher, dem Codex Einsidlensis aus dem 10. Jahrhundert, befindet sich folgende Aufgabe: Ein Schiff mit 30 Passagieren, je zur Hälfte Christen und Türken, droht wegen Überlastung zu sinken. Um zu entscheiden, welche Hälfte über Bord muß, stellen sich die Passagiere in einen Kreis und sondern jeden 10. aus. Wie müssen sich die Christen in den Kreis stellen, damit sie an Bord bleiben können?

Diese nicht unbedingt von christlicher Nächstenliebe zeugende Aufgabe heißt Josephus-Problem nach dem jüdischen Historiker Josephus Flavius (37 - 100). Nach eigenem Bericht war er mit einem Trupp von 40 Widerstandskämpfern bei einem Aufstand im Jahre 67 von den Römern eingeschlossen worden. Um dem allgemein beschlossenen Selbstmord zu entgehen, schlug er vor, daß sich jeder 10. selbst töten sollte. Er stellte sich dann so in den Kreis, daß er zusammen mit einem Freund übrig blieb.
Durch Abzählen ermittelt man die Lösung.



Josephus und sein Freund mußten sich an die 19. und 35. Position stellen.

Zum folgenden Programm

Die Anzahl der Leute und die Schrittweite werden eingegeben, die Aussonderung nach diesen Kriterien wird dann ausgedruckt (siehe Ergebnisausdruck).

```
100 REM JOSEPHUS-PROBLEM
110 :
120 DIM N(100)
130 PRINT"JOSEPHUS-PROBLEM"
140 PRINT "AUS WIEVIELEN LEUTEN WIRD AUSGESONDERT";
150 INPUT K
160 PRINT "MIT WELCHER SCHRITTWEITE WIRD AUSGESONDERT";
170 INPUT S
180 :
190 REM INITIALISIEREN
200 FOR I=1 TO K-1
210 N(I)=I+1
220 NEXT I
230 N(K)=1
240 :
250 REM AUSSONDERN
260 PRINT"IN DIESER REIHENFOLGE WIRD AUSGESONDERT:"
270 Z=K
280 FOR I=1 TO S-1
290 Z=N(Z)
300 NEXT I
310 PRINT N(Z);
320 N(Z)=N(N(Z))
330 IF N(Z)<>Z THEN 280
340 :
350 PRINT Z
360 PRINT""
370 END
READY.
```

```
JOSEPHUS-PROBLEM

AUS WIEVIELEN LEUTEN WIRD AUSGESONDERT? 41

MIT WELCHER SCHRITTWEITE WIRD AUSGESONDERT? 10

IN DIESER REIHENFOLGE WIRD AUSGESONDERT:

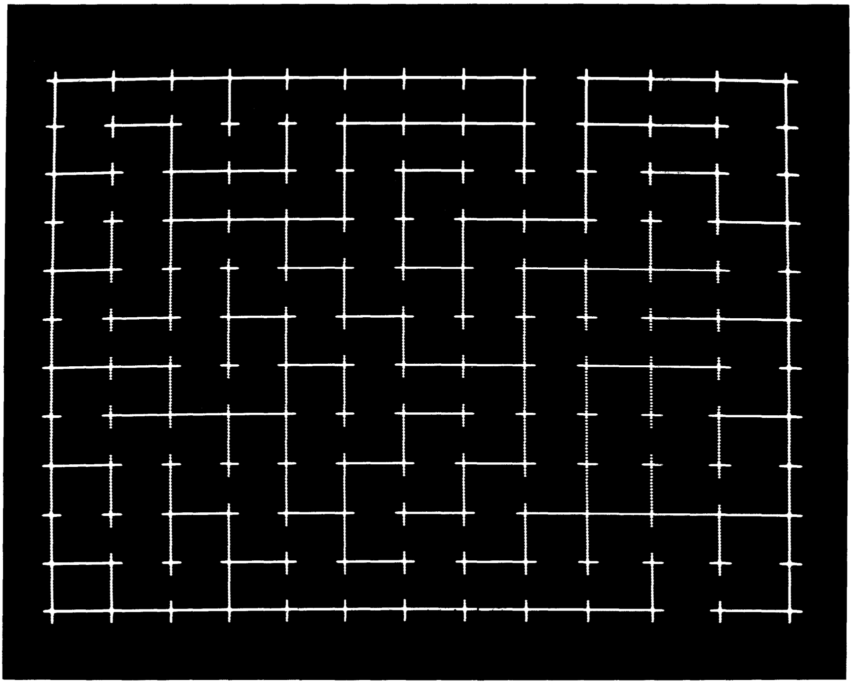
10,20,30,40,09,21,32,02
14,26,38,11,24,37,12,27
01,17,34,08,29,06,28,07
33,16,41,25,18,05,03,29
04,15,23,13,36,22,31,19,35
```

16. Labyrinth

Wer kennt sie nicht, die Sage vom Ungeheuer Minotaurus auf Kreta, das in einem unentrinnbaren Labyrinth hauste? Nur dem Wollknäuel der Königstochter Ariadne verdankte es Theseus, daß er dort wieder herausfand.

Labyrinthe sind in der Architektur keine Seltenheit. Herodot berichtete sogar von einem ägyptischen Labyrinth, das angeblich 3000 Kammern besaß. Bekannt sind auch die englischen Schloßgärten in Form von Labyrinthen. Das berühmteste Labyrinth ist wohl der 1690 für William of Orange entworfene Park von Hampton Court Palace, in dem noch heute die Touristen auf die Hecken steigen, um nach dem Ausgang zu suchen.

Labyrinthe werden in der Verhaltensforschung zur Messung von Intelligenz von Säugetieren verwendet. Auch die Lernfähigkeit der kybernetischen "Maus", 1951 von Shannon gebaut, wurde an einem Labyrinth getestet. Hat sie einmal aus einem Irrgarten herausgefunden, wird sie dies immer wieder tun können.



Zum folgenden Programm

Das Programm konstruiert nach Eingabe der gewünschten Länge und Breite ein Labyrinth mit genau einem Durchgang. Wenn der Cursor in der linken oberen Ecke erscheint, kann die Graphik durch Drücken der Space-Taste abgerufen werden. Zu beachten ist noch, daß bei maximaler Größe am rechten Bildschirmrand ein Zeilenvorschub erfolgt; in diesem Fall sollte die Zeile 1500 gelöscht werden.
Ein ähnliches Programm, das auch noch zusätzlich einen Weg durch das Labyrinth sucht, findet sich in [2] unter dem Namen "Maze".

```
100 REM LABYRINTH
110 :
120 PRINT"       SPIELANLEITUNG (J/N)";
130 INPUT A$:IF A$="J"THEN 150
140 GOTO 200
150 PRINT"DAS PROGRAMM KONSTRUIERT EIN LABYRINTH"
160 PRINT"MIT GENAU EINEM DURCHGANG.WENN DER"
170 PRINT"CURSOR IN DER LINKEN OBEREN ECKE"
180 PRINT"ERSCHEINT,DRUECKEN SIE DIE SPACE-TASTE"
190 :
200 PRINT"":INPUT"LAENGE (L<=13) UND BREITE (B<=11)";L,B
210 IFL>2 AND B>2 THEN 230
220 GOTO200
230 PRINT"       MOMENT BITTE,COMPUTER UEBERLEGT"
240 :
250 DIMF(L,B),T(L,B)
260 Q=0:Z=0:X=1
270 GOTO350
280 FORI=1TOL
290 IFI=XTHEN310
300 PRINT"+--";:GOTO320
310 PRINT"+  ";
320 NEXTI
330 PRINT"+"
340 RETURN
350 C=1:W(X,1)=C:C=C+1
```

```
360 J=X:K=1:GOTO430
370 IFJ<>LTHEN410
380 IFK<>BTHEN400
390 J=1:K=1:GOTO420
400 J=1:K=K+1:GOTO420
410 J=J+1
420 IFF(J,K)=0THEN370
430 IFJ-1=0THEN760
440 IFF(J-1,K)<>0THEN760
450 IFK-1=0THEN590
460 IFF(J,K-1)<>0THEN590
470 IFJ=LTHEN510
480 IFF(J+1,K)<>0THEN510
490 X=INT(RND(1)*3+1)
500 ONXGOTO1100,1140,1180
510 IFK<>BTHEN540
520 IFZ=1THEN570
530 Q=1:GOTO550
540 IFF(J,K+1)<>0THEN570
550 X=INT(RND(1)*3+1)
560 ONXGOTO1100,1140,1250
570 X=INT(RND(1)*2+1)
580 ONXGOTO1100,1140
590 IFJ=LTHEN690
600 IFF(J+1,K)<>0THEN690
```

```
610 IFK<>BTHEN640
620 IFZ=1THEN670
630 Q=1:GOTO650
640 IFF(J,K+1)<>0THEN670
650 X=INT(RND(1)*3+1)
660 ONXGOTO1100,1180,1250
670 X=INT(RND(1)*2+1)
680 ONXGOTO1100,1180
690 IFK<>BTHEN720
700 IFZ=1THEN750
710 Q=1:GOTO730
720 IFF(J,K+1)<>0THEN750
730 X=INT(RND(1)*2+1)
740 ONXGOTO1100,1250
750 GOTO1100
760 IFK-1=0THEN950
770 IFF(J,K-1)<>0THEN950
780 IFJ=LTHEN880
790 IFF(J+1,K)<>0THEN880
800 IFK<>BTHEN830
810 IFZ=1THEN860
820 IFQ=1GOTO840
830 IFF(J,K+1)<>0THEN860
840 X=INT(RND(1)*3+1)
850 ONXGOTO1140,1180,1250
860 X=INT(RND(1)*2+1)
```

```
870 ONXGOTO1140,1180
880 IFK<>BTHEN910
890 IFZ=1THEN940
900 Q=1:GOTO920
910 IFF(J,K+1)<>0THEN940
920 X=INT(RND(1)*2+1)
930 ONXGOTO1140,1250
940 GOTO1140
950 IFJ=LTHEN1040
960 IFF(J+1,K)<>0THEN1040
970 IFK<>BTHEN1000
980 IFZ=1THEN1030
990 Q=1:GOTO1150
1000 IFF(J,K+1)<>0THEN1030
1010 X=INT(RND(1)*2+1)
1020 ONXGOTO1180,1250
1030 GOTO1180
1040 IFK<>BTHEN1070
1050 IFZ=1THEN1090
1060 Q=1:GOTO1080
1070 IFF(J,K+1)<>0THEN1090
1080 GOTO1250
1090 GOTO1370
1100 F(J-1,K)=C
```

```
1110 C=C+1:T(J-1,K)=2:J=J-1
1120 IFC=L*B+1THEN1380
1130 Q=0:GOTO430
1140 F(J,K-1)=C
1150 C=C+1
1160 T(J,K-1)=1:K=K-1:IFC=L*B+1THEN1380
1170 Q=0:GOTO430
1180 F(J+1,K)=C
1190 C=C+1:IFT(J,K)=0THEN1210
1200 T(J,K)=3:GOTO1220
1210 T(J,K)=2
1220 J=J+1
1230 IFC=L*B+1THEN1380
1240 GOTO760
1250 IFQ=1THEN1310
1260 F(J,K+1)=C:C=C+1:IFT(J,K)=0THEN1280
1270 T(J,K)=3:GOTO1290
1280 T(J,K)=1
1290 K=K+1:IFC=L*B+1THEN1380
1300 GOTO430
1310 Z=1
1320 IFT(J,K)=0THEN1350
1330 T(J,K)=3:Q=0
1340 GOTO1370
1350 T(J,K)=1:Q=0
1360 J=1:K=1:GOTO420
```

```
1370 GOTO370
1380 POKE32768,160
1390 WAIT158,1:GETU$
1400 PRINT"❐"
1410 GOSUB280
1420 FORM=1TOB
1430 PRINT"|";
1440 FORI=1TOL
1450 IFT(I,M)<2THEN1480
1460 PRINT"   ";
1470 GOTO1490
1480 PRINT"  |";
1490 NEXTI
1500 PRINT
1510 FORI=1TOL
1520 IFT(I,M)=0THEN1560
1530 IFT(I,M)=2THEN1560
1540 PRINT"+  ";
1550 GOTO1570
1560 PRINT"+——";
1570 NEXTI
1580 PRINT"+"
1590 NEXTM
1600 CLR
1610 GOTO 200
READY.
```

17. Magisches Quadrat I

Magische Quadrate gehören wohl zu den ältesten Objekten mathematischer Bemühungen. Nach einem Bericht des Kung-fu-tse erschien das magische Quadrat Lo-Shu,

4	9	2
3	5	7
8	1	6

dem chinesischen Kaiser Yü (ca. 2200 v. Chr.) auf dem Rücken der heiligen Schildkröte Hi. Seit altersher erregten die magischen Quadrate die Phantasie der Menschen: sie wurden als astrologische Offenbarung bestaunt, mit Zahlenmystizismus in Verbindung gebracht und sogar bis in die Neuzeit als Amulette gegen alle möglichen Krankheiten verordnet (sogar noch von Paracelsus).

Bekannt ist auch das magische Quadrat in Dürers Holzschnitt "Melancholia" 1514:

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

Dies magische Quadrat ist besonders interessant, da die Summe der Zahlen in den fünf eingezeichneten Teilquadraten gleich der magischen Zahl des Quadrates selbst ist.

In unserer aufgeklärten Zeit haben die magischen Quadrate natürlich ihren Mystizismus verloren. Sie bilden aber weiterhin einen Gegenstand mathematischer Forschung, mit dem sich schon so berühmte Mathematiker wie Arthur Cayley und Oswald Veblen beschäftigten.

Für das Folgende ist vorausgesetzt, daß es sich um ein echtes magisches Quadrat handelt, also um ein die Zahlen 1, 2, 3, . . ., n^2 umfassendes Quadrat, in dem jede Zeile, Spalte und Diagonale stets dieselbe Summe, magische Zahl genannt, ergibt. Die Anzahl der Zahlen in einer Zeile oder Spalte nennt man die Ordnung des Quadrats.

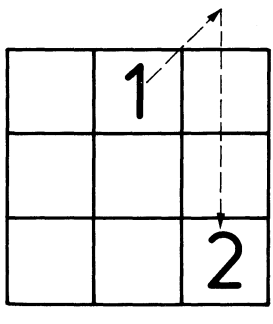
Die magische Zahl eines Quadrats der Ordnung n kann leicht berechnet werden, sie ist nämlich die Summe aller Zahlen dividiert durch die Ordnung des Quadrats:

$$\frac{1 + 2 + 3 + \ldots + n^2}{n} = \frac{(n^2+1)\, n^2}{2n} = \frac{1}{2}\,(n^3+n)$$

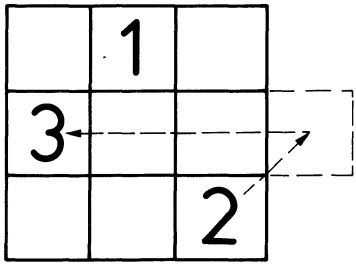
Die Ordnung des Lo-Shu ist somit 15 und die des Dürerquadrats 34. Mit Hilfe der angegebenen Formel läßt sich die magische Zahl für beliebig große Quadrate ermitteln. Weitere Literatur findet sich in [10] oder [6].

Im folgenden soll nun ein Verfahren angegeben werden, das die Konstruktion magischer Quadrate ungerader Ordnung gestattet. Es stammt von dem Jesuiten Simon de la Loubere, der die Methode 1687 aus China mitbrachte. Zwei weitere Verfahren finden sich in [10].

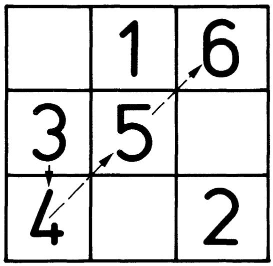
Das Verfahren von la Loubere wird am Quadrat der Ordnung 3 gezeigt:
Ausgehend von der Mitte der 1. Zeile, geht man in das nächste Kästchen rechts oben (Normalschritt). Da dieses Kästchen außerhalb liegt, geht man nach unten in die letzte Zeile und schreibt die "2".



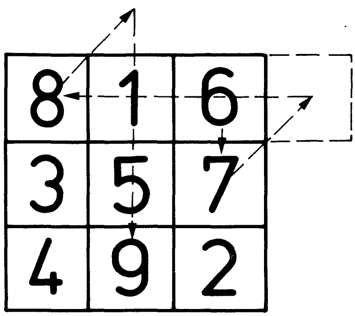
Da der Normalschritt von der "2" ausgeführt, wieder zu einem Kästchen außerhalb führt, schreibt man die "3" in die Mitte der ersten Spalte.



Da von "3" aus der Normalschritt nicht durchführbar ist, macht man einen Notschritt senkrecht nach unten und schreibt die "4" unter die "3". In Normalschritten ergibt sich ebenfalls die "5" und "6".



Da von "6" aus der Normalschritt nicht möglich ist, setzt man im Notschritt die "7" darunter. Die "8" rutscht wie die "3" nach rechts und die "9" wie die "2" nach unten.



Man sieht, daß das Quadrat durch Spiegelung an der waagrechten Achse in das Lo-Shu übergeht.

Zum folgenden Programm

Nach Eingabe einer ungeraden Zahl wird nach dem Algorithmus von la Loubere das zugehörige magische Quadrat berechnet. Für Ordnung 9 liefert das Programm:

47	58	69	80	1	12	23	34	45
57	68	79	9	11	22	33	44	46
67	78	8	10	21	32	43	54	56
77	7	18	20	31	42	53	55	66
6	17	19	30	41	52	63	65	76
16	27	29	40	51	62	64	75	5
26	28	39	50	61	72	74	4	15
36	38	49	60	71	73	3	14	25
37	48	59	70	81	2	13	24	35

Die magische Zahl ist 369.

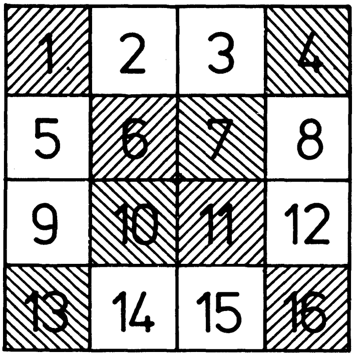
```
100 REM MAGISCHES QUADRAT UNGERADER ORDNUNG
110 :
120 INPUT"ORDNUNG DES QUADRATS";N
130 IF(N+1)/2<>INT(N+1)/2 THEN 120
140 DIM M(N,N)
150 :
160 REM ANFANGSWERTE
170 I=1:K=1:Z=1
180 J=(N+1)/2
190 :
200 REM BERECHNEN DER MATRIXELEMENTE
210 M(I,J)=Z
220 Z=Z+1
230 IF Z>N*N THEN 420
240 REM NACH N SCHRITTEN NEUE DIAGONALE
250 IF K<N THEN 300
260 K=1
270 I=I+1
280 GOTO 210
290 REM NORMALSCHRITT
300 K=K+1
310 I=I-1
320 J=J+1
330 REM RAND OBEN
340 IF I<>0 THEN 380
350 I=N
```

```
360 GOTO 210
370 REM RAND RECHTS
380 IF J<=N THEN 210
390 J=1
400 GOTO 210
410 :
420 REM AUSGABE
430 PRINT"[illegible]MAGISCHES QUADRAT DER ORDNUNG[illegible]";N;"[illegible]"
440 FOR I=1 TO N
450 FOR J=1 TO N
460 PRINTM(I,J);
470 NEXTJ
480 PRINT
490 NEXTI
READY.
```

18. Magisches Quadrat II

Nicht ganz so einfach sind die Verfahren zur Erzeugung von magischen Quadraten gerader Ordnung.
So war auch Benjamin Franklin sehr stolz auf seine 1769 veröffentlichte Liste magischer Quadrate gerader Ordnung. Besonders freute er sich über sein magisches Quadrat der Ordnung 16; es sei das magischste von allem Magischen, das je ein Magier hervorgebracht habe, schrieb er darüber euphorisch. Jedoch war sein Quadrat nicht ganz magisch, da die Diagonalensumme nicht die magische Zahl 2056 ergab.

Das hier benützte Diagonalverfahren setzt voraus, daß die Ordnung des Quadrats durch 4 teilbar ist. Andere Verfahren findet man in [10]. Es soll hier für n = 4 gezeigt werden:
Ein Quadrat mit aufsteigender Numerierung wird in 4 2x2-Quadrate zerlegt.



Von diesen Teilquadraten werden jeweils 2 Diagonalelemente markiert bzw. schraffiert. Tauscht man je zwei bezüglich des Mittelpunkts symmetrisch liegende Felder aus, so erhält man ein magisches Quadrat der Ordnung 4.

16	2	3	13
5	11	10	8
9	7	6	12
4	14	15	1

Wie man sieht, geht das so erhaltene magische Quadrat durch Vertauschung der beiden mittleren Spalten in das Dürer-Quadrat über.

Zum folgenden Programm

Nach Eingabe der durch 4 teilbaren Ordnung berechnet das Programm das zugehörige magische Quadrat nach der Diagonalmethode.
Zunächst werden die Diagonalfelder durch Vorzeichenwechsel markiert und dann entsprechend ausgetauscht.
Für n = 8 liefert das Programm folgendes magische Quadrat:

64	2	62	4	5	59	7	57
9	55	11	53	52	14	50	16
48	18	46	20	21	43	23	41
25	39	27	37	36	30	34	32
33	31	35	29	28	38	26	40
24	42	22	44	45	19	47	17
49	15	51	13	12	54	10	56
8	58	6	60	61	3	63	1

Die magische Zahl ist 260.

```
100 REM MAGISCHES QUADRAT DER ORDNUNG 4*K
110 :
120 INPUT"ORDNUNG D.MAG.QUADRATS";N
130 IF N/4<>INT(N/4)THEN PRINT"ORDNUNG MUSS TEILER 4 HABEN":GOTO 120
140 PRINTCHR$(147)
150 PRINT"█MAGISCHES QUADRAT DER ORDNUNG";N:PRINT
160 DIM A%(N↑2)
170 :
180 REM INITIALISIEREN
190 N1=N/2:MA=1:MI=N/2:U=N↑2+1:N2=N↑2/2
200 :
210 REM MARKIEREN DES QUADRATS
220 FOR K=1 TO N1
230 FOR J=1 TO N1
240 A%(MI-J+1)=MA:A%(MI+J)=MA:MA=-MA
250 NEXT J
260 MI=MI+N:MA=-MA
270 NEXT K
280 :
290 REM AUSTAUSCHEN DER MARKIERTEN FELDER
300 FOR I=1 TO N2
310 L=I
320 IF A%(I)<0 THEN L=U-I
330 A%(L)=I:A%(U-L)=U-I
340 NEXT I
350 :
```

```
 360 REM AUSGABE
 370 FOR I=1 TO N↑2
 380 PRINTA%(I);
 390 IF I/N=INT(I/N) THEN PRINT
 400 NEXT I
READY.
```

19. Magisches Quadrat III

Nach den beiden vorangegangenen Programmen fehlt nur noch ein Verfahren für Quadrate mit gerader Ordnung, das nicht durch 4 teilbar ist.
Eine solche Methode ist das Torusverfahren (andere Verfahren finden sich in [10]). Es soll hier für n = 6 demonstriert werden:

Das gesuchte Quadrat wird aus vier gleichgroßen Teilquadraten ungerader Ordnung zusammengesetzt. Das erste ist das an der waagrechten Achse gespiegelte la Loubere-Quadrat der Ordnung 3:

4	9	2
3	5	7
8	1	6

Die anderen drei Teilquadrate erhält man, indem man zu allen Zahlen des ersten jeweils 9 oder 18 bzw. 27 addiert:

13	18	11
12	14	16
17	10	15

22	27	20
21	23	25
26	19	24

31	36	29
30	32	34
35	28	33

Ineinandersetzen dieser 4 Teilquadrate liefert schon fast das gesuchte Quadrat

4	22	9	27	2	20
31	13	36	18	29	11
3	21	5	23	7	25
30	12	32	14	34	16
8	26	1	19	6	24
35	17	28	10	33	15

Vertauscht man je zwei übereinanderstehende schraffierte Felder, so erhält man das gesuchte magische Quadrat der Ordnung 6:

31	22	9	27	2	20
4	13	36	18	29	11
3	21	32	23	7	25
30	12	5	14	34	16
35	26	1	19	6	24
8	17	28	10	33	15

Zum folgenden Programm

Nach Eingabe der Ordnung wird zunächst geprüft, ob die Zahl gerade, aber nicht durch 4 teilbar ist. In einem Unterprogramm, das viermal aufgerufen wird, werden die vier Teilquadrate erzeugt. Schließlich werden die entsprechenden Felder ausgetauscht und das fertige Quadrat ausgegeben. Bei Eingabe von n = 10 liefert das Programm folgendes Quadrat:

86	36	93	68	25	75	2	52	9	59
11	61	18	43	100	50	77	27	84	34
85	35	87	62	19	69	21	71	3	53
10	60	12	37	94	44	96	46	78	28
79	29	6	56	88	63	20	70	22	72
4	54	81	31	13	38	95	45	97	47
98	48	80	55	7	57	14	64	16	66
23	73	5	30	82	32	89	39	91	41
92	42	99	74	1	51	8	58	15	65
17	67	24	49	76	26	83	33	90	40

Die magische Zahl ist 505.

```
100 REM MAGISCHES QUADRAT DER ORDNUNG 4K+2
110 :
120 INPUT"ORDNUNG DES QUADRATS";N
130 IF (N-INT(N/4)*4)=2 THEN 160
140 PRINT"ORDNUNG D.QUADRATS MUSS BEI DER DIVISON"
150 PRINT" DURCH 4 DEN REST 2 LASSEN !":GOTO 120
160 M=N/2:N1=N:N=N-1
170 REM TEILQUADRATE AUFSTELLEN
180 FOR P=0 TO 3
190 T=P
200 GOSUB 390
210 NEXT P
220 REM KAESTCHEN AUSTAUSCHEN
230 FOR H=0 TO M-1
240 FOR J=0 TO M-3
250 IF (H=INT(M/2)) AND (J=M-3) THEN 270
260 V=A(2*H,J):A(2*H,J)=A(2*H+1,J):A(2*H+1,J)=V:GOTO 280
270 V=A(M-1,M-1):A(M-1,M-1)=A(M,M-1):A(M,M-1)=V
280 NEXT J
290 NEXT H
300 REM AUSGABE
310 PRINT"[illegible]MAGISCHES QUADRAT ORDNUNG";"[illegible]";N1;"[illegible]"
320 FOR I=0 TO N
330 FOR K=0 TO N
340 PRINT A(I,K);
350 NEXT K
```

```
 360 PRINT
 370 NEXT I
 380 END
 390 I=N-3+T-INT(T/2)*2
 400 IF T/3=INT(T/3) THEN K=M-3:GOTO 420
 410 K=M-2
 420 Z=T*M↑2
 430 FOR H=0 TO M-1
 440 FOR J=0 TO M-1
 450 I=(I+2)-INT((I+2)/N1)*N1
 460 K=(K+2)-INT((K+2)/N1)*N1
 470 Z=Z+1
 480 A(I,K)=Z
 490 NEXT J
 500 I=I-4
 510 K=K-2
 520 NEXT H
 530 RETURN
READY.
```

20. Logelei

Folgende Logelei soll nun mit Hilfe des Computers gelöst werden:
Fünf Leute wollen unter folgenden Bedingungen zu einer Party kommen:

(1) Wenn A nicht kommt, dann D
(2) B kommt nur gemeinsam mit D oder gar nicht
(3) Wenn A kommt, dann auch C und D
(4) Wenn C kommt, dann auch E
(5) B kommt, wenn E nicht kommt und umgekehrt.

In Programmiersprachen wie PASCAL oder FORTRAN kann man direkt mit Wahrheitswerten rechnen. Die Wahrheitswert-Tafeln lauten:

A	B	Nicht A	A oder B	A und B
w	w	f	w	w
w	f	f	w	f
f	w	w	w	f
f	f	w	f	f

wobei w für wahr und f für falsch steht. Damit sind aber noch nicht alle Verknüpfungsmöglichkeiten erschöpft. Es fehlen noch $A \Rightarrow B$ (Aus A folgt B), $A \equiv B$ (A und B haben gleichen Wahrheitswert) und $A \not\equiv B$ (A und B haben ungleichen Wahrheitswert):

A	B	$A \Rightarrow B$	$A \equiv B$	$A \not\equiv B$
w	w	w	w	f
w	f	f	f	w
f	w	w	f	w
f	f	w	w	f

Mit Hilfe eines Tricks kann man im CBM-BASIC trotzdem mit Wahrheitswerten rechnen.
Die Zahl "0" wird maschinenintern als

0000 0000 0000 0000

codiert. Durch Anwendung des Operators NOT werden alle Binärstellen (Bits) invertiert, d.h. NOT 0 wird zu

1111 1111 1111 1111.

Dies ist genau die interne Darstellung von −1. Somit kann man falsch mit "0" und wahr mit "−1" gleichsetzen, wenn man überprüft, ob die Verknüpfungen OR und AND die Wahrheitswert-Tafeln erfüllen. Dies ist tatsächlich der Fall.

Wie man leicht einsieht, können auch die übrigen Verknüpfungen wie folgt codiert werden:

$A \Rightarrow B$	A>=B
$A \equiv B$	A = B
$A \not\equiv B$	A<>B.

Die fünf Bedingungen unserer Logelei können nun so codiert werden:

(1) (NOT A)>=D

(2) B=D

(3) A>=(C AND D)

(4) C>=E

(5) B<>E.

Da nun alle fünf Bedingungen gleichzeitig gelten müssen, ist der Wahrheitswert-Term

((NOT A)>= D) AND (B=D) AND (A>=(C AND D)) AND (C>=E) AND (B<>E).

Genau wenn dieser Term wahr ist, d.h. −1 ist, sind alle fünf Bedingungen erfüllt.

Mit Hilfe von fünf verschachtelten Schleifen der Art

FOR X = −1 TO 0

werden alle möglichen Kombinationen von den Wahrheitswerten A,B,C,D und E erzeugt. Läßt man jetzt alle Kombinationen ausdrucken, bei denen der obere Wahrheitswert-Term den Wert −1 hat, erhält man die Lösung der anfangs erwähnten Logelei.

Das Programm liefert

0 −1 0 −1 0;

es gibt also nur eine Lösung mit A = falsch, B = wahr, C = falsch, D = wahr und E = falsch. Es kommen also nur B und D zur Party.

Alle anderen Logeleien dieser Art können damit ebenfalls gelöst werden.

```
100 REM LOGELEI
110 :
120 REM 5 LEUTE WOLLEN UNTER FOLGENDEN BEDINGUNGEN ZU EINER PARTY KOMMEN:
130 REM  WENN A NICHT KOMMT,DANN D
140 REM  B KOMMT NUR MIT D ODER GAR NICHT
150 REM  WENN A KOMMT,DANN AUCH C UND D
160 REM  WENN C KOMMT,DANN AUCH E
170 REM  B KOMMT,WENN E NICHT KOMMT UND UMGEKEHRT
180 :
190 FOR A=-1 TO 0
200 FOR B=-1 TO 0
210 FOR C=-1 TO 0
220 FOR D=-1 TO 0
230 FOR E=-1 TO 0
240 S=((NOTA)>=D)AND(B=D)AND(A>=(C AND D))AND(C>=E)AND(B<>E)
250 IF S=-1 THEN PRINTA;B;C;D;E
260 NEXT E
270 NEXT D
280 NEXT C
290 NEXT B
300 NEXT A
310 END
READY.
```

```
LOGELEI                    0    -1     0    -1     0
```

21. Wer war der Täter?

Eine weitere bekannte Art von Logikaufgaben besteht darin, Schlußfolgerungen aus mehreren, z.T. widersprüchlichen Angaben unter der Bedingung zu ziehen, daß nur eine der Aussagen richtig ist.

Ein Beispiel dafür:
Ein Kommissar hat vier Verdächtige verhaften lassen: Pistolen-Heini, Quassel-Joe, Räuber-Maxe und Spelunken-Ede. Während des Verhörs machen die vier folgende Aussagen:

(1) Quassel-Joe ist der Täter
(2) Spelunken-Ede ist der Täter
(3) Räuber-Maxe ist nicht der Täter
(4) Spelunken-Ede ist nicht der Täter

Wen soll der Kommissar verhaften lassen, wenn er weiß, daß genau eine der vier Aussagen wahr ist?

Wie bei Programm Nr. 20 ausgeführt, können die Wahrheitswerte wahr durch "−1" und falsch durch "0" codiert werden. Addiert man nun vier solcher Wahrheitswerte, so erkennt man, daß die Summe genau dann −1 (also wahr) ist, wenn genau ein Summand −1 (d.h. wahr) ist.

Nimmt man nun nacheinander jeweils eine der folgenden Annahmen:

"Pistolen-Heini war der Täter"
"Quassel-Joe war der Täter"
"Räuber-Maxe war der Täter"
"Spelunken-Ede war der Täter"

als wahr an, so kann der Wahrheitswert der oben erwähnten Summe bestimmt werden. Ist dieser Wahrheitswert für alle vier Annahmen genau einmal −1, so gibt es eine eindeutige Lösung und derjenige, dessen Wahrheitswert gerade −1 (wahr) ist, ist der Täter.
Ist der Wahrheitswert der Summe niemals −1, so gibt es keine Lösung. Tritt der Wahrheitswert −1 mehrfach auf, so gibt es mehrere Verdächtige und somit keine eindeutige Lösung.

Zum folgenden Programm

Die vier verdächtigen Personen werden durch das Feld P (1) – Pistolen-Heini, P (2) – Quassel-Joe, P (3) – Räuber-Maxe und P (4) – Spelunken-Ede codiert. In einer Schleife wird jeweils genau ein Wert auf –1 (d.h. wahr) gesetzt, die anderen entsprechend auf Null. Die Aussagen (1) bis (4) werden zur folgenden Summe zusammengefaßt:

P (2) + P (4) + NOTP (3) + NOTP (4).

Der Zähler J bestimmt, wie oft diese Summe den Wert –1 annimmt. Der Index L weist auf den Täter hin, falls eine eindeutige Lösung existiert. Entsprechend dem Wert des Zählers erfolgt die Ausgabe der Lösung.
Das Programm liefert L = 3, somit ist "Räuber-Maxe" der Täter.

Mit der hier angegebenen Methode können alle Logikaufgaben, bei denen genau eine Bedingung wahr ist, gelöst werden.

```
100 REM WER WAR DER TAETER?
110 :
120 REM FOLGENDE KNOBELEI WIRD GELOEST:
130 REM  PISTOLEN-HEINI(P),QUASSEL-JOE(Q)
140 REM  RAEUBER-MAXE(R),SPELUNKEN-EDE(S)
150 REM  WURDEN VERHAFTET.IHRE AUSSAGEN SIND:
160 REM        Q WAR ES
170 REM        S WAR ES
180 REM        R WAR ES NICHT
190 REM        S WAR ES NICHT
200 :
210 REM DER KOMMISSAR WEISS:GENAU EINE DER AUSSAGEN IST RICHTIG.
220 REM   WEN KANN ER VERHAFTEN ?
230 :
240 J=1:L=0
250 FOR I=1 TO 4
260 FOR K=1 TO 4:P(K)=0:NEXTK
270 P(I)=-1
280 IF P(2)+P(4)+(NOTP(3))+(NOTP(4))=-1 THEN J=J+1:L=I
290 NEXT I
300 :
310 ON J GOTO 330,340,350
320 END
330 PRINT "KEINE LOESUNG GEFUNDEN":END
340 PRINT MID$("PQRS",L,1);" WAR ES":END
350 PRINT "ES GIBT MEHRERE VERDAECHTIGE".
360 PRINT"WEITERE INFORMATIONEN NOTWENDIG."
370 END
READY.
```

22. Kryptogramm

Bekannt von den Wochenend-Ausgaben vieler Zeitungen sind die Zahlenrätsel oder Kryptogramme, wobei alle Ziffern einer Rechenaufgabe eindeutig durch einen Buchstaben oder ein Symbol dargestellt werden.

Ein bekanntes Beispiel dafür liefert

$$\begin{array}{r} \text{S E N D} \\ +\ \text{M O R E} \\ \hline \text{M O N E Y} \end{array}$$

Es ist nicht schwer herauszufinden, daß (vgl. [6])

$$\begin{array}{r} 9\,5\,6\,7 \\ +\ 1\,0\,8\,5 \\ \hline 1\,0\,6\,5\,2 \end{array}$$

die einzige mögliche Lösung ist. Daß Kryptogramme durchaus mehrere Lösungen haben können, zeigt folgendes Beispiel:

$$\begin{array}{r} \text{W I R E} \\ +\ \text{M O R E} \\ \hline \text{M O N E Y} \end{array}$$

Es gibt, wie man leicht nachprüfen kann, nicht weniger als fünf Lösungen:

$$\begin{array}{r} 9\,7\,6\,2 \\ +\ 1\,0\,6\,2 \\ \hline 1\,0\,8\,2\,4 \end{array}$$

$$\begin{array}{r} 9\,2\,7\,4 \\ +\ 1\,0\,7\,4 \\ \hline 1\,0\,3\,4\,8 \end{array}$$

$$\begin{array}{r} 9\,5\,7\,4 \\ +\ 1\,0\,7\,4 \\ \hline 1\,0\,6\,4\,8 \end{array}$$

$$\begin{array}{r} 9\,2\,8\,7 \\ +\ 1\,0\,8\,7 \\ \hline 1\,0\,3\,7\,4 \end{array}$$

$$\begin{array}{r} 9\,5\,8\,7 \\ +\ 1\,0\,8\,7 \\ \hline 1\,0\,6\,7\,4 \end{array}$$

Zum folgenden Programm

Wie man sieht, steht M für 1, da die Summe zweier vierstelliger Zahlen höchstens 19.998 betragen kann. Innerhalb zweier Schleifen werden für D und E alle möglichen Werte zwischen 2 und 9 erzeugt. Y ist dann die Summe von D und E bzw. der Zehnerrest. Entsprechend wird der Übertrag U1 der Einerstelle bestimmt. Da R unbestimmt ist, durchläuft es ebenfalls eine Schleife von 2 bis 9. N ergibt sich damit aus der Addition der Zehnerziffern, entsprechend der Übertrag U2. Da nun N bestimmt ist, läßt sich O und der Übertrag U3 aus der Summe der Hunderterziffern ermitteln. Da M bereits bekannt ist, läßt sich S aus der Addition der Tausenderziffern berechnen.

Wichtig ist, daß durch die Abfragen

IF R=Y OR R=E OR R=D THEN ....

verhindert wird, daß zwei Buchstaben dieselbe Ziffer darstellen.

Auf diese Art können auch andere Kryptogramme mit Hilfe des Computers entziffert werden.

```
100 REM KRYPTOGRAMM
110 :
120 PRINT"GESUCHT SIND ALLE LOESUNGEN DES KRYPTOGRAMMS:"
130 PRINTTAB(12)"S E N D"
140 PRINTTAB(10)"+ M O R E"
150 PRINTTAB(9)"____________"
160 PRINTTAB(10)"M O N E Y"
180 PRINT"DABEI STEHT JEDER BUCHSTABE FUER EINE"
190 PRINT"BESTIMMTE ZIFFER."
200 PRINT""
210 :
220 REM ALLE VARIABLENBEZEICHNUNGEN ENTSPRECHEN DEM KRYPTOGRAMM
230 M=1
240 FOR D=2 TO 9
250 FOR E=2 TO 9
260 IF E=D THEN 480
270 Y=E+D
280 IF Y>=10 THEN U1=1:Y=Y-10:GOTO 300
290 U1=0
300 IF Y=M THEN 480
310 FOR R=2 TO 9
320 IF R=Y OR R=E OR R=D THEN 470
330 IF E>=R+U1 THEN U2=0:N=E-R-U1:GOTO 350
340 U2=1:N=10+E-R-U1
350 IF N=R OR N=E OR N=Y OR N=D OR N=M THEN 470
360 IF N>=E+U2 THEN U3=0:O=N-E-U2:GOTO 380
370 U3=1:O=10+N-E-U2
380 IF O=N OR O=R OR O=E OR O=Y OR O=D OR O=M THEN 470
```

```
390 IF O>M+U2-1 THEN 470
400 S=O+10-M-U3
410 IF S=O OR S=N OR S=R OR S=E OR S=Y OR S=D OR S=M THEN 470
420 PRINTTAB(12)S;E;N;D
430 PRINTTAB(11)"+";M;O;R;E
440 PRINTTAB(10)"_____________"
450 PRINTTAB(9)M;O;N;E;Y
460 PRINT
470 NEXT R
480 NEXT E
490 NEXT D
500 END
READY.
```

23. Das Jeep-Problem

Ein Jeepfahrer muß unter folgenden Bedingungen eine Wüste durchqueren:

(1) Die einzige Tankmöglichkeit ist ein Tanklager am Rande der Wüste.

(2) Der Jeep kann nur soviel Benzin aufladen, wie sein Tank und seine Reservekanister fassen.

Gesucht ist die größtmögliche Strecke, die der Jeep unter diesen Bedingungen zurücklegen kann.

Ist T die Tankfüllung des Jeeps und V der Benzinverbrauch, so kann der Jeep mit einer Füllung

$$\frac{T}{V} \cdot 100 \text{ km}$$

zurücklegen. Zur Überwindung einer größeren Strecke müssen daher Vorratsdepots in der Wüste angelegt werden.
Die optimale Vorgehensweise ist wohl, den gesamten Benzinvorrat vom Startpunkt A zum Zwischenlager B zu fahren, entsprechend von B nach C und so fort.

Ist der Benzinvorrat anfangs das k+1-fache des Tankinhalts, so muß der Jeep

(k+1) mal von A nach B

und

k mal von B nach A

fahren. Ist E_1 die Entfernung AB, so legt er insgesamt die Strecke

$$(2k+1) \cdot E_1 \cdot V$$

zurück. Da er dafür nur eine Tankfüllung brauchen darf, gilt

$$(2k+1) \cdot E_1 \cdot V = T.$$

Die zurückgelegte Entfernung ergibt sich somit zu

$$E_1 = \frac{T}{(2k+1) \cdot V}$$

Wegen der abnehmenden Benzinvorräte gilt nun für die Entfernung BC analog

$$E_2 = \frac{T}{(2k-1) \cdot V}$$

und so fort.

Die Summe aller dieser Entfernungen ist somit

$$E = \frac{T}{V} \cdot \left(\frac{1}{2k+1} + \frac{1}{2k-1} + \frac{1}{2k-3} + \ldots\ldots + \frac{1}{3} + 1\right).$$

An den Zwischenlagern kann der Fahrer jeweils soviel Treibstoff zurücklassen, wie von einer Tankfüllung vom zweimaligen Zurücklegen der Strecke E_1, E_2 usw. übrigbleibt. Dies liefert die Zwischenlager-Menge

$$T - 2 \cdot V \cdot E_i \quad (i = 1, 2, 3 \ldots.)$$

Damit ist das Problem vollständig gelöst.

Zum folgenden Programm

Die Entfernung E_i und die in den Zwischenlagern zurückgelassenen Benzinmengen werden mit Hilfe zweier Funktionen

```
FND(X) = T/((2*X+1)*V)
FNZ(X) = T - 2*V*FND(X)
```

berechnet. Der Gesamtbenzinvorrat B, der Tankinhalt T und der Benzinverbrauch V wird in Form von DATA-Werten eingelesen. Innerhalb einer Schleife wird der verbleibende Benzinvorrat, die zurückgelegte Distanz und die Benzinmenge des Vorratslagers berechnet. Durch Aufsummieren der einzelnen Distanzen wird die Gesamtdistanz ermittelt.

Für die Daten

Gesamt-Benzinvorrat B = 320 l
Tankinhalt des Jeeps T = 40 l
Benzinverbrauch V = 8,2 l je 100 km

liefert das Programm die Gesamtdistanz von 986,24 km (vergleiche Bildschirm-Ausdruck).

```
100 REM JEEP-PROBLEM
110 :
120 READ B : REM BENZINVORRAT
130 READ T : REM TANKINHALT
140 READ V : REM BENZINVERBRAUCH JE 100 KM
150 :
160 REM BERECHNUNG DER DISTANZ
170 DEF FND(X)=T/((2*X+1)*V)
180 :
190 REM BERECHNUNG DES ZWISCHENLAGERS
200 DEF FNZ(X)=T-2*V*FND(X)
210 :
220 REM RUNDUNG
230 DEF FNR(X)=INT(100*X+.5)/100
240 :
250 K=B/T : REM ANZAHL DER TANKVORGAENGE
260 PRINT"■JEEP-PROBLEM■"
270 PRINT"GESAMT-BENZINVORRAT";B
280 PRINT"■TANKINHALT DES JEEPS";T
290 PRINT"■BENZINVERBRAUCH JE 100 KM";V
300 :
310 PRINT"■FAHRTNR.";TAB(9)"BENZ.VORR.";TAB(20)"DISTANZ";TAB(29)"ZW.LAGER"
320 PRINT
330 G=0
340 FOR J=1 TO K-1
350 B=B-T
360 D=100*FND(K-J):G=G+D
370 Z=FNZ(K-J)
380 PRINT J;TAB(10)B;TAB(20)FNR(D);TAB(29)FNR(Z)
```

```
390 NEXT J
400 :
410 REM LETZTE FAHRT
420 D=100*FND(0):G=G+D:B=B-T
430 PRINTK;TAB(10)B;TAB(20)FNR(D);TAB(30)"0"
440 PRINT"QGESAMTDISTANZ=";FNR(G)
450 END
460 :
470 DATA 320
480 DATA 40
490 DATA 8.2
READY.
```

JEEP-PROBLEM

GESAMT-BENZINVORRAT 320

TANKINHALT DES JEEPS 40

BENZINVERBRAUCH JE 100 KM 8.2

FAHRTNR.	BENZ.VORR.	DISTANZ	ZW.LAGER
1	280	32.52	34.67
2	240	37.52	33.85
3	200	44.35	32.73
4	160	54.2	31.11
5	120	69.69	28.57
6	80	97.56	24
7	40	162.6	13.33
8	0	487.8	0

GESAMTDISTANZ= 986.24

24. Das Kokosnuß-Problem

In der Ausgabe der "Saturday Evening Post" vom 9.10.1926 erschien eine Kurzgeschichte von Ben Amen Williams mit dem Titel "Kokosnüsse".
Die Geschichte handelt von einem Bauunternehmer, der einem Konkurrenten einen wichtigen Bauabschluß vor der Nase wegschnappen konnte, weil dieser sich nur noch mit einem Rätsel beschäftigte. Dieses Rätsel war das inzwischen berühmte Kokosnuß-Problem.

Hier das Rätsel, wie es in Williams Geschichte erzählt wird (vgl. [4]): Fünf Seeleute und ein Affe werden durch einen Schiffbruch auf eine einsame Insel verschlagen und verbringen den ersten Tag mit Kokosnuß-Sammeln. Aus Angst, am nächsten Morgen bei der Teilung zu kurz zu kommen, steht jeder der Männer in der Nacht auf, nimmt sich ein Fünftel der jeweils vorhandenen Nüsse und gibt eine, bei der Teilung übrigbleibende, Nuß dem Affen. Am Morgen teilen sie die restlichen Nüsse in fünf gleiche Teile. Wieviele Kokosnüsse waren zu Beginn vorhanden?
Williams unterließ es, seiner Geschichte die Lösung beizufügen und so wurde die Zeitungsredaktion förmlich mit Leserbriefen überschwemmt, die nach der richtigen Lösung fragten. Der Chefredakteur sandte daraufhin Williams folgendes Telegramm: "Um Gottes Willen, wieviele Kokosnüsse? Hier ist die Hölle los!".

Sind A, B, C, . . ., F die Anteile, die bei den sechs Teilungen auftreten, so ergeben sich folgende ganzzahlige Gleichungen (auch diophantisch) genannt:

$N = 5A + 1$	1. Teilung
$4A = 5B + 1$	2. Teilung
$4B = 5C + 1$	3. Teilung
$4C = 5D + 1$	4. Teilung
$4D = 5E + 1$	5. Teilung
$4E = 5F.$	letzte Teilung

Dabei ist N die ursprüngliche Zahl der Kokosnüsse.

Diese Gleichungen können in einer Gleichung zusammengefaßt werden:

$$1024 \cdot N = 15625 \cdot F + 8404.$$

Eine solche diophantische Gleichung hat aber im allgemeinen mehrere, meist sogar unendlich viele Lösungen. Man beschränkt sich daher oft auf die Suche nach der kleinsten Lösung.

Diese ist N = 3121 und F = 204. Daß dies eine Lösung ist, sieht man relativ schnell; nicht so einfach ist der Nachweis, daß dies tatsächlich die kleinstmögliche Lösung ist:

$$
\begin{aligned}
3121 &= 5 \cdot 624 + 1 \\
4 \cdot 624 &= 5 \cdot 499 + 1 \\
4 \cdot 499 &= 5 \cdot 399 + 1 \\
4 \cdot 399 &= 5 \cdot 319 + 1 \\
4 \cdot 319 &= 5 \cdot 255 + 1 \\
4 \cdot 255 &= 5 \cdot 204.
\end{aligned}
$$

Es waren anfangs also 3121 Kokosnüsse, der erste Mann holte sich 624, der zweite 499, der dritte 399, der vierte 319 und der fünfte 255 Kokosnüsse. Von den am Morgen verbleibenden 1020 Stück erhielt jeder noch einmal 204 Nüsse.
Wegen der großen Zahlen, die bei diesem diophantischen Problem auftreten, ist die Lösung nicht leicht zu finden. Kein Wunder also, wenn die Redaktion der "Saturday Evening Post" mit Leserbriefen überschüttet wurde.

Erwähnenswert ist auch die Lösung des Problems, wenn der Affe am Morgen bei der 6. Teilung ebenfalls eine Nuß abbekommt. Hier ist die kleinste ganzzahlige Lösung

$$15621.$$

Interessant ist dabei, daß die zugehörige diophantische Gleichung

$$1024 \; N = 15625 \; F + 11529$$

mit Hilfe von negativen Zahlen (!) gelöst werden kann (siehe [4]).

Zum folgenden Programm

In einer Schleife werden die Zahlen 6, 11, 16, 21, 26, . . . usw. erzeugt und geprüft, ob die oben erwähnten sechs Teilungen ganzzahlig durchgeführt werden können. Ist dies der Fall, so wird die entsprechende Zahl ausgedruckt, ansonsten wird weitergezählt. Der Computer findet die Lösung, wenn er bis 3121 gezählt hat. Die obengenannte Lösung 15621 erhält man, wenn man die Grenze für T in Zeile 300 um eins erhöht. Die Rechenzeit ist dann entsprechend größer.

```
100 REM KOKOSNUSS-PROBLEM
110 :
120 REM  5 SEELEUTE UND 1 AFFE SAMMELN EINEN VORRAT KOKOSNUESSE
130 REM  IN DER NACHT STEHT JEDER DER MAENNER AUF UND NIMMT
140 REM  1/5 DER VORHANDENEN NUESSE UND GIBT 1 UEBRIGBLEIBENDE
150 REM  NUSS DEM AFFEN.AM NAECHSTEN MORGEN WERDEN DIE
160 REM  RESTLICHEN NUESSE AUF DIE 5 MAENNER VERTEILT.
170 :
180 REM  GESUCHT IST DIE KLEINSTE ANZAHLEN VON NUESSEN
190 REM  DIE DIESE TEILUNG ZULAESST.
200 :
210 REM H : RESTL.HAUFEN V.NUESSEN
220 REM T : NUMMER DER TEILUNG
230 REM V : URSPRUENGL.VORRAT
240 :
250 V=6
260 :
270 T=0:H=V
280 H=(H-1)*4/5
290 T=T+1
300 IF T<5 AND INT(H)=H THEN 280
310 IF INT(H)=H THEN 340
320 V=V+5:GOTO 270
330 :
340 PRINT"ES WAREN URSPRUENGLICH";V;"KOKOSNUESSE"
350 END
READY.
```

25. Wäge-Problem

Wäge-Aufgaben gibt es in zahllosen Variationen:
Sie unterscheiden sich in der Zahl der Kugeln, der Art der Waage und den Angaben, ob die gesuchte Kugel schwerer oder leichter ist.
Bei der hier gestellten Aufgabe soll mit einer Balkenwaage und möglichst wenig Wiegevorgängen eine schwere Kugel aus einer beliebigen Zahl von sonst gleichartigen Kugeln ermittelt werden.

Ein optimales Vorgehen besteht darin, daß die Anzahl der Kugeln zuerst in drei Drittel geteilt werden. Legt man dann je ein Drittel der Kugeln auf die Balkenwaage, so kann dasjenige Drittel bestimmt werden, das die schwerere Kugel enthält. Bleibt die Waage im Gleichgewicht, so befindet sich die gesuchte Kugel im nicht auf der Waage liegenden Drittel.
Senkt sich aber die Waage auf eine Seite, so liegt die zu bestimmende Kugel in dieser Waagschale. Mit dem so bestimmten Drittel setzt man die Drittelung solange fort, bis die gesuchte Kugel gefunden ist.

Zum folgenden Programm

Nach Eingabe der Anzahl der Kugeln und der Nummer der schwereren Kugel, erhalten alle Kugeln mit Ausnahme der gesuchten, das Gewicht 1, die gedachte das Gewicht 2. Nun werden die Kugeln gedrittelt und gewogen. Dies geschieht, indem man die Gewichte der Kugeln in den entsprechenden Indexgrenzen addiert. Entsprechend wird die Länge des Index-Intervalls gedrittelt. Das Verfahren setzt sich in der angegebenen Weise fort, bis die Untergrenze U des Index-Intervalls mit der Obergrenze O übereinstimmt. Die Waagschale enthält dann nur noch eine Kugel, nämlich die gesuchte.

```
100 REM WAEGE-PROBLEM
110 :
120 REM  VON N GLEICHAUSSEHENDEN KUGELN SOLL MIT HILFE
130 REM  EINER BALKENWAAGE MIT MOEGLICHST WENIG WAEGUNGEN
140 REM  DIE KUGEL ERMITTELT WERDEN,DIE SCHWERER
150 REM IST ALS DIE ANDEREN.
160 :
170 REM K(I) : KUGELN
180 REM W : ZAHL DER WAEGUNGEN
190 REM U : UNTERGRENZE DES INDEX
200 REM O,O1,O2 : OBERGRENZE DES INDEX
210 REM G1,G2 : GEWICHTE
220 :
230 INPUT"WIEVIELE KUGELN";N
240 DIM K(N)
250 FOR I=1 TO N
260 K(I)=1
270 NEXT I
280 INPUT"WELCHE KUGEL SOLL ES SEIN";J
290 K(J)=2
300 W=0:U=1:O=N
310 :
320 L=INT((O+2-U)/3)
330 O1=U+L-1
340 O2=O1+L
350 G1=0:G2=0
```

```
360 FOR I=U TO O1
370 G1=G1+K(I)
380 NEXT I
390 FOR I=O1+1 TO O2
400 G2=G2+K(I)
410 NEXT I
420 W=W+1
430 IF G1=G2 THEN U=O2:GOTO 460
440 IF G1>G2 THEN O=O1:GOTO 460
450 U=O1+1:O=O2
460 IF O<>U THEN 320
470 :
480 PRINT"■DIE GESUCHTE KUGEL WAR DIE";U;".TE"
490 PRINT"ES WAREN";W;"WAEGUNGEN NOTWENDIG"
500 END
READY.
```

26. Extremwert

Als Beispiel einer Extremwert-Aufgabe sollen die Ausmaße einer zylindrischen 1-Liter-Dose so bestimmt werden, daß möglichst wenig Weißblech notwendig ist.
Wie man anschaulich sieht, erfüllt eine sehr flache Dose z.B. der Höhe 1 cm nicht die Anforderungen, da ihre Grundfläche dann 1000 cm^2 betragen muß. Ebenso kann eine sehr hohe Dose, z.B. vom Radius 1 cm, ausgeschlossen werden, da ihre Höhe dann 3,18 m betragen würde. Da beide Fälle nicht auf minimalen Materialverbrauch schließen lassen, liegen die gesuchten Maße der Dose in einem mittleren Bereich.

Unter der Annahme, daß es einen kleinsten Wert für die Oberfläche der Dose gibt, kann man diesen auch ohne Differentialrechnung finden:
Man fängt mit einem kleinen Radius an und berechnet jeweils den Materialverbrauch. Dann erhöht man jeweils den Radius um eine kleine Schrittweite und vergleicht den neuen Materialverbrauch mit dem alten Wert. Da wir angenommen haben, dem kleinsten Wert näher zu kommen, wird der Materialaufwand zunächst kleiner werden. Ist jedoch das Minimum überschritten, so wird der Materialverbrauch wieder größer werden. An dieser Stelle kann die Rechnung dann abgebrochen werden.

Zum folgenden Programm

Das Programm startet bei einem Radius R von 2 cm. In einer Schleife wird der Radius jeweils um 1 mm erhöht und die zugehörige Oberfläche der Dose berechnet. Am Ende der Schleife wird die neu berechnete Oberfläche mit dem bisherigen minimalen Wert verglichen. Die Schleife wird verlassen, wenn die neue Oberfläche wieder größer wird. Es werden dann der vorhergegangene Radius und die entsprechenden Maße der Dose bestimmt.

Das Programm liefert die Werte R = 5,4 cm, H = 10.9 cm und Oberfläche O = 553,6 cm^2.
Diese Werte stimmen ziemlich genau mit den exakten Ergebnissen

$$R = 5{,}42 \text{ cm}, \quad H = 10{,}84 \text{ cm}, \quad O = 553{,}58 \text{ cm}^2$$

überein. Interessant ist dabei, daß der Durchmesser der idealen Dosenform mit der Höhe übereinstimmt.

```
100 REM EXTREMWERT
110 :
120 REM ES WERDEN DIE AUSMASSE EINER ZYLINDRISCHEN 1 L-DOSE
130 REM MIT MINIMALEM MATERIALVERBRAUCH BESTIMMT
140 :
150 REM V : VOLUMEN
160 REM O : OBERFLAECHE
170 REM H : HOEHE
180 REM M : MANTELFLAECHE
190 REM G : GRUNDFLAECHE
200 REM R : RADIUS
210 DEF FNR(X)=INT(10*X+.5)/10 : REM RUNDUNG
220 :
230 V=1000 : R=2 : O=99999
240 REM SUCHSCHLEIFE
250 R=R+.1
260 MIN=O
270 G=π*R↑2
280 H=V/G
290 M=2*π*R*H
300 O=2*G+M
310 IF O<= MIN THEN 250
320 R=R-.1
330 H=V/π/R↑2
340 :
350 PRINT "DER MINIMALE MATERIALBEDARF IST";FNR(MIN);"CM↑2"
360 PRINT"RADIUS=";FNR(R);"CM"
370 PRINT"HOEHE=";FNR(H);"CM"
380 END
READY.
```

27. Ulamsche Vermutung

Eines von den bekannteren, noch ungelösten mathematischen Problemen ist die sog. Ulamsche Vermutung. Er nahm an, daß das folgende Verfahren, auf beliebige natürliche Zahlen angewandt, stets bei der Zahl 1 aufhört.

(1) Wähle eine beliebige Zahl A

(2) Ist A = 1, dann fertig.

(3) Ist A gerade, so ersetze A durch A/2. Gehe zu (2)

(4) Ist A ungerade, so ersetze A durch 3·A+1. Gehe nach (2)

Diese Vermutung ist bis heute unbewiesen, obwohl D.E. Selfridge von der Universität Berkeley mit Hilfe von Computern gezeigt hat, daß die Ulamsche Vermutung bis

$$2^{29} = 536.870.912$$

richtig ist.

Zum folgenden Programm

Nach Eingabe der Zahl wird die nach dem obigen Verfahren erzeugte Zahlenfolge ausgedruckt, bis sie mit 1 endet. Bei der Eingabe von großen Zahlen muß beachtet werden, daß bei Zahlen über 32768 der Ganzzahlbereich des Computers verlassen wird, es können dann Rundungsfehler bei der Division auftreten.
Für die Zahl 127 ist der Bildschirm-Ausdruck angegeben.

```
100 REM ULAMSCHE VERMUTUNG
110 :
120 REM FOLGENDES VERFAHREN WIRD AUF EINE NATUERLICHE ZAHL ANGEWANDT:
130 REM IST DIE ZAHL GERADE,SO WIRD SIE HALBIERT
140 REM IST SIE UNGERADE,SO WIRD SIE VERDREIFACHT UND EINS ADDIERT
150 REM AUF DIE SO ENTSTEHENDE ZAHL,WIRD DAS VERFAHREN ERNEUT ANGEWANDT
160 REM BIS DIE ZAHL EINS ERSCHEINT.
170 REM ULAM VERMUTETE,DASS DAS VERFAHREN FUER BELIEBIGE
180 REM ZAHLEN STETS BEI EINS ENDET
190 :
200 INPUT"GIB ZAHL EIN";N
210 PRINT"DIE ZAHLENFOLGE IST:"
220 PRINT N;
230 :
240 IF N/2=INT(N/2) THEN N=N/2;GOTO 260
250 N=3*N+1
260 PRINT N;
270 IF N<> 1 THEN 240
280 END
READY.
```

```
.
GIB ZAHL EIN? 127
DIE ZAHLENFOLGE IST:
 127  382  191  574  287  862  431 1294  647  1942  971  2914  1457  4372  2186
  1093  3280  1640  820  410  205  616  308  154  77  232  116  58  29  88  44
22  11  34  17  52  26  13  40  20  10  5  16  8  4  2  1
```

28. Verschlüsselung

Die Wissenschaft vom Verschlüsseln von Nachrichten – Kryptologie genannt – hat zu allen Zeiten große Bedeutung gehabt.
Man hat schon in Uruk, der frühen Hauptstadt Babyloniens, verschlüsselte Keilschrift-Tafeln aus dem 3. vorchristlichen Jahrhundert gefunden. Der systematische Gebrauch von Geheimschriften begann um 1200 im Vatikan.

Die von dem Franzosen Blaise de Vigenere 1582 erfundene Schlüsselwort-Methode galt fast dreihundert Jahre als unentschlüsselbar, bis 1863 ein preußischer Major namens Kasiski dafür eine methodische Lösung fand. Noch nicht möglich war die Entschlüsselung, wenn man anstatt eines Schlüsselwortes einen fortlaufenden Text wählte. Diese Methode wandte die "Rote Kapelle", eine kommunistische Untergrundorganisation im 2. Weltkrieg an. Als es während des Kriegs dem englischen Mathematiker Turing gelang, den Code der reichsdeutschen Marine zu knacken, kam es zu einer entscheidenden Kriegswende.
Ersetzt man nach einem Vorschlag des amerikanischen Nachrichtentechnikers G.S. Vernan den fortlaufenden Text durch eine Folge von Zufallszahlen, so erhält man eine Codierung, die nicht zu brechen ist. Dies folgt aus der Tatsache, daß eine Reihe von Zufallszahlen keinerlei Regelmäßigkeit unterliegt und damit auch nicht vorhersagbar ist. So trug z.B. 1957 der berühmte Sowjetagent Rudolf Abel bei seiner Verhaftung Zufallszahlen-Tabellen bei sich. Alle neueren Verschlüsselungs-Methoden beruhen letztlich auf der Vernan-Chiffrierung.

Die Schlüsselwort-Methode hat folgendes Prinzip: Die Buchstaben des Alphabets werden durchnumeriert mit A = 1, B = 2, C = 3 usw.

Addiert man nun zu den Nummern der Nachricht

C O M P U T E R	=	3	15	13	16	21	20	5	18

die des Schlüsselwortes

B A S I C/B A S	=	2	1	19	9	3	2	1	19
so ergibt sich die Verschlüsselung:		5	16	32 =6	25	24	22	6	37 =11
E P F Y X V F K									

Ist das Schlüsselwort zu kurz, so beginnt man das Wort von neuem. Ist die Summe der Buchstaben-Nummern größer als 26, so nimmt man den Rest, der bei der Division durch 26 bleibt.

Zum folgenden Programm

Das Programm fragt zuerst ab, ob verschlüsselt oder entschlüsselt werden soll (Eingabe "1" oder "2").
Dann wird die zu verschlüsselnde Nachricht und das Schlüsselwort abgefragt.
Die Umrechnung von Buchstaben in Nummern erfolgt nach dem ASCII-Code:
A = 65, B = 66, C = 67 usw.
Wegen der zusätzlichen Codierung der Leerstellen werden im Programm die Summen der Buchstaben-Nummern um 27 reduziert.

```
100 REM VERSCHLUESSELUNG
110 :
120 DIM L(100),K(100)
130 PRINT"GIB ZU VERSCHLUESSELNDE NACHRICHT EIN!"
140 INPUT M$
150 L=LEN(M$)
160 PRINT"GIB SCHLUESSELWORT EIN!"
170 INPUT K$
180 K=LEN(K$)
190 PRINT"VERSCHLUESSELN 1 ODER ENTSCHLUESSELN 2"
200 INPUT A
210 IF A<>1 AND A<>2 THEN 190
220 :
230 FOR J=1 TO K
240 K(J)=ASC(MID$(K$,J,1))-64
250 IF A=1 THEN 270
260 K(J)=27-K(J)
270 NEXT J
280 :
290 REM UMWANDELN
300 J=1
310 FOR I=1 TO L
320 X=ASC(MID$(M$,I,1))
330 X=X+K(J)
340 IF X<=ASC("Z") THEN 360
350 X=X-27
```

```
360 L(I)=X
370 J=J+1
380 IF J<=K THEN 400
390 J=1
400 NEXT I
410 :
420 REM AUSGABE
430 PRINT"███DIE NACHRICHT LAUTET"
440 FOR I=1 TO L
450 IF L(I)=59 THEN L(I)=32
460 PRINTCHR$(L(I));
470 NEXT I
480 PRINT
490 END
READY.
```

29. Informatik – Quiz

Das Programm stellt ein Quiz mit 50 Fragen zur Geschichte der Informatik dar. Diese Fragen betreffen hauptsächlich die Anfänge der Rechentechnik, Erfinder von Rechenmaschinen, Erbauer von Computern und Programmiersprachen. Die Fragen und zugehörigen Antworten werden in Form von DATA-Werten eingelesen. Durch eine Zufallszahl wird ein stets wechselnder Anfang hergestellt.

```
100 REM INFORMATIK-QUIZ
110 :
120 X=INT(49*RND(1)+1)
130 FOR I=1 TO X
140 READ A$,B$
150 NEXT I
160 X=X+1:IF X=50 THEN RESTORE
170 READ A$,B$
180 PRINT"INFORMATIK-QUIZ"
190 PRINT"";A$
200 PRINT"":INPUT"ANTWORT(KEINE=0)";C$
210 IF C$=B$ THEN PRINT"RICHTIG !":GOTO 240
220 IF MID$(C$,1,4)=MID$(B$,1,4) THEN PRINT"GENAUER",B$:GOTO 240
230 PRINT"RICHTIG IST:",B$
240 FOR I=1 TO 700:NEXT I
250 GOTO 160
260 DATA WELCHES VOLK ERFINDET UM -600 DAS DEZIMALSYSTEM,INDER
270 DATA WIE IST DER ROEMISCHE NAMEN DES RECHENBRETTS UM -500,ABAKUS
280 DATA WELCHER SPAETERE PABST BAUT UM 1000 EINEN ABAKUS,GERBERT
290 DATA WANN SCHRIEB AL-KWARISMI SEIN 1.RECHENBUCH IM DEZIMALSYSTEM,825
300 DATA WELCHER BEGRIFF LEITET SICH VOM NAMEN AL-KWARISMI AB,ALGORITHMUS
310 DATA WER SCHRIEB 1202 DAS 1.RECHENBUCH MIT INDISCHEN ZIFFERN,FIBONACCI
320 DATA WIE LAUTETE DER EIGENTLICHE NAME DES FIBONACCI,LEONARDO VON PISA
330 DATA WER HATTE 1275 DIE IDEE EINER DENKMASCHINE,LULLUS
340 DATA WER GIBT 1614 DIE 1.LOGARITHMENTAFEL HERAUS,NAPIER
350 DATA WELCHES GERAET (1620)RECHNET EBENFALLS LOGARITHMISCH,RECHENSCHIEBER
```

```
 360 DATA WER ERBAUTE 1623 DIE ERSTE RECHENMASCHINE,SCHICKARD
 370 DATA WER ERBAUTE 1642 DIE 2.RECHENMASCHINE,PASCAL
 380 DATA WELCHER ENGLAENDER ERFINDET GLEICH 3 RECHENMASCHINEN,MORLAND
 390 DATA WER ERFINDET 1673 EINE VIER-SPEZIES-RECHENMASCHINE,LEIBNIZ
 400 DATA WER ERFINDET 1679 DAS BINAERSYSTEM,LEIBNIZ
 410 DATA WER BAUTE 1770 DIE 1.SERIENMAESSIGE RECHENMASCHINE,HAHN
 420 DATA WER BAUTE 1728 EINEN LOCHKARTEN-GESTEUERTEN WEBSTUHL,FALCON
 430 DATA WER STELLTE 1801 SERIENMAESSIG LOCHKARTEN-WEBSTUEHLE AUS,JACQUARD
 440 DATA WER ENTWARF 1823 DIE DIFFERENCE ENGINE ZUR TAFELWERKSBERECHNUNG,BABBAG
E
 450 DATA WANN ENTWARF BABBAGE DIE ANALYTICAL ENGINE,1835
 460 DATA WER ALGEBRAISIERTE 1854 DIE GESETZE DER LOGIK,BOOLE
 470 DATA WER ERFAND 1884 DIE LOCHKARTENMASCHINE,HOLLERITH
 480 DATA IN WELCHEM LAND WURDE 1890 LOCHKARTEN ZUR VOLKSZAEHLUNG EINGESETZT,USA
 490 DATA WANN WURDEN DIE 1.ELEKTR.VERSTAERKERROEHREN GEBAUT,1916
 500 DATA WER ENTWICKELTE 1938 DIE SCHALTALGEBRA,SHANNON
 510 DATA WER GRUENDETE 1948 DIE KYBERNETIK,WIENER
 520 DATA WER SCHUF 1936 DIE THEORIE DER BERECHENBARKEIT,TURING
 530 DATA WANN ERFANDEN BARDEEN BRATTAIN UND SHOCKELEY DEN TRANSISTOR,1948
 540 DATA WER ENTWARF 1934 DEN 1.MECHANISCHEN COMPUTER,ZUSE
 550 DATA WANN LIEF DER 1.FUNKTIONIERENDE COMPUTER,1941
 560 DATA WELCHE SCHALTUNGEN BENUETZTE DER Z3 VON KONRAD ZUSE,RELAIS
 570 DATA WER BAUTE 1942 DEN BELL RELAY INTERPOLATOR,STIBITZ
 580 DATA WIE HIESS DER ENTWURF EINER PROGRAMMIERSPRACHE VON ZUSE 1945,PLANKALKU
EL
 590 DATA WER BAUTE 1944 DEN RELAIS-RECHNER MARK 1,AIKEN
```

```
 600 DATA WER HATTE 1945 DIE IDEE DER SPEICHER-PROGRAMMIERUNG,VON NEUMANN
 610 DATA WER BAUTE 1946 DEN 1.ELEKTRONENROEHREN-COMPUTER,ECKERT/MAUCHLY
 620 DATA WER BAUTE DEN 1.PROGRAMMGESTEUERTEN COMPUTER EDSAC,WILKES
 630 DATA WIE HIESS DER 1.SERIENMAESSIG GEFERTIGTE COMPUTER,UNIVAC 1
 640 DATA WER ENTWICKELTE 1954 DIE PROGRAMMIERSPRACHE FORTRAN,BACKUS
 650 DATA WER BAUTE 1958 DIE 1.INTEGRIERTE SCHALTUNG,KILBY
 660 DATA WANN KAM DER 1.VOLLTRANSISTORISIERTE COMPUTER AUF DEN MARKT,1959
 670 DATA WELCHE TECHN.NATURWISS.PROGRAMMIERSPRACHE WURDE 1960 GENORMT,ALGOL 60
 680 DATA WELCHE KAUFMAENN.PROGRAMMIERSPRACHE WURDE 1961 IN USA EINGEFUEHRT,COBO
L
 690 DATA WER ERFAND DIE PROGRAMMIERSPRACHE PASCAL,WIRTH
 700 DATA WER ENTWICKELTE 1964 DIE PROGRAMMIERSPRACHE BASIC,KEMENY/KURTZ
 710 DATA WIE HEISST DIE WEITERENTWICKLUNG VON ALGOL 60,ALGOL 68
 720 DATA WANN KAMEN DIE 1.TASCHENRECHNER AUF DEN MARKT,1973
 730 DATA WANN KAMEN DIE 1.HEIMCOMPUTER AUF DEN MARKT,1978
 740 DATA WIE HEISST DIE 1976 ENTWICKELTE UNTERRICHTS-PROGRAMMIERSPRACHE,ELAN
 750 DATA WELCHE PROGRAMMSPRACHE IST NACH ADA AUGUSTA OF LOVELACE GENANNT,ADA
READY.
```

30. Promille-Test

Der folgende Promille-Test gehört eigentlich nicht zu den Knobeleien, kann aber auch spaßeshalber betrieben werden.
Nach Eingabe des entsprechenden Trink-Quantums wird zunächst die Menge des konsumierten Alkohols in Gramm berechnet. Nach Eingabe des Körpergewichts wird dann der Blutalkoholspiegel ermittelt. Für jede verflossene Stunde nach dem letzten Glas können Sie noch 0,11 Promille vom Ergebnis abziehen.

```
100 REM PROMILLE-TEST
110 :
120 A$="■■■■■■■■■■■■■■■■"
130 FOR I=1 TO 50
140 PRINTA$;"PROMILLETEST"
150 PRINTA$;"■PROMILLETEST■"
160 NEXT I
170 PRINT"■■■GEBEN SIE IHR KOERPERGEWICHT IN KG EIN !"
180 INPUT M
190 PRINTA$;
200 INPUT"WIEVIELE HALBE BIER";B
210 A=B*.05*.8*500:PRINT
220 PRINTA$;
230 INPUT"WIEVIEL■ STAMPERL SCHNAPS";S
240 A=A+S*.45*.8*20:PRINT
250 PRINTA$;
260 INPUT"WIEVIELE STAMPERL LIKOER";L
270 A=A+L*.28*.8*20:PRINT
280 PRINTA$;
290 INPUT"WIEVIELE SCHOPPEN WEIN";W
300 A=A+W*.10*.8*250:PRINT
310 PRINTA$;
320 INPUT"WIEVIELE GLAESER SEKT";C
330 A=A+C*.10*.8*150:PRINT
340 DEFFNF(X)=INT(100*X)/100
350 A=FNF(A)
```

```
360 PRINTA$;"SIE HABEN"
370 PRINT"████████████";A;"██GRAMM"
380 PRINT"██████████";"REINEN ALKOHOL GETRUNKEN"
390 FOR I=1 TO 1000:NEXT I
400 P=A/(.7*M)
410 PRINTA$;"DAS MACHT BEI IHNEN"
420 PRINT"█████████";FNF(P);"█PROMILLE"
430 PRINT"█████"
440 END
READY.
```

31. Münzwurf

Das Nachvollziehen eines Zufallsexperiments – wie z.B. Würfeln oder Münzen-Werfen – mit Hilfe von Zufallszahlen, nennt man Monte-Carlo-Simulation. Die Methode wurde 1944 von den Mathematikern Von Neumann und Ulam eingeführt, um das Verhalten von Neutronen im Inneren eines Atom-Reaktors oder einer Bombe zu studieren. Aber das schon 1777 beschriebene Buffon-Nadelproblem (siehe Programm 33) kann als typisches Monte-Carlo-Experiment angesehen werden. Interessant ist, daß die Monte-Carlo-Methoden auch zur Lösung von nicht-statistischen Aufgaben – wie z.B. der Integralrechnung – herangezogen werden können.

Eine Zufallszahl ist eine zufällig ausgewählte Zahl x, für die gilt $0 \leqslant X < 1$. Folgen von Zufallszahlen werden mit Computern nach Verfahren, die streng geheim gehalten werden, errechnet. Da die nächstfolgende aus der vorhergehenden berechnet wird, handelt es sich streng genommen um sog. Pseudo-Zufallszahlen. Die Bedeutung der Zufallszahlen liegt darin, daß mit ihrer Hilfe beliebige Wahrscheinlichkeitsmodelle nachgespielt werden können. So regellos und zufällig Zufallszahlen auch sind, jedes Verfahren zur Erzeugung von Zufallszahlen unterliegt genauen statistischen Tests.

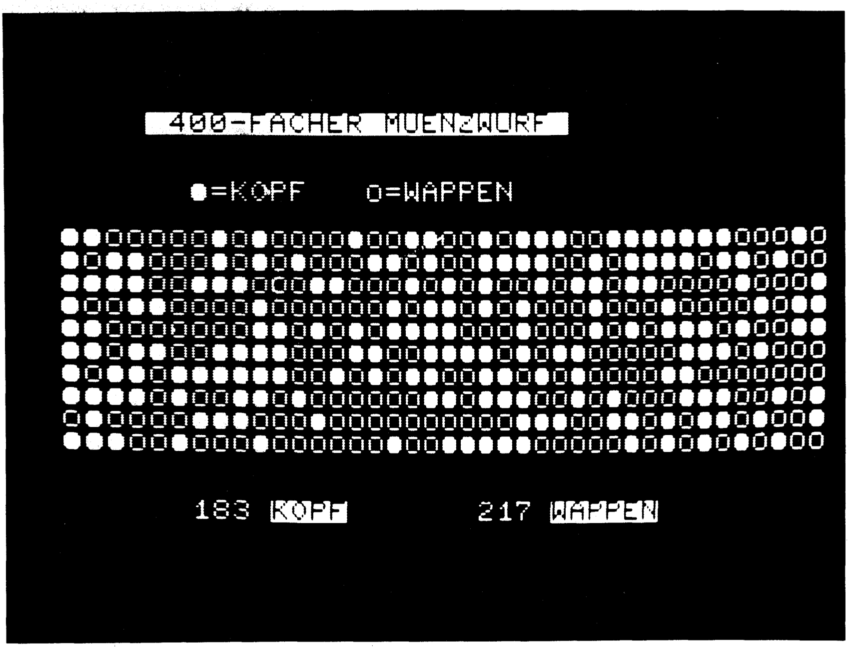
In BASIC ist ein Zufallszahlen-Generator standardmäßig implementiert. Er wird durch den Befehl RND(X) (Random Digit) aufgerufen, bei jedem Aufruf mit $x > 0$ erzeugt er eine neue Zufallszahl. Ein Münzwurf kann nun leicht simuliert werden: Ist die Zufallszahl kleiner als 0,5, so zeigt die Münze "Kopf", ansonsten "Wappen". Würfelergebnisse können wie folgt erhalten werden:

INT(RND(1) * 6 + 1)

der INT-Befehl sorgt durch Abrundung für ganze Zahlen (siehe z.B. Programm 6).

Zum folgenden Programm

Nach Eingabe der gewünschten Wurfanzahl werden "Kopf" oder "Wappen" am Bildschirm graphisch dargestellt (siehe Programmausdruck). Gleichzeitig wird die jeweilige Anzahl der Ergebnisse eingeblendet.



```
100 REM MUENZWURF
110 :
120 INPUT"WIEVIELE WUERFE";N
130 PRINT"█████████";N;"█-FACHER MUENZWURF "
140 PRINT"███████ ●=KOPF    O=WAPPEN█"
150 :
160 W=0
170 FOR I=1 TO N
180 IF RND(1)<.5 THEN PRINT"●";:GOTO 200
190 PRINT"o";:W=W+1
200 NEXT I
210 :
220 PRINT:PRINT"██████";N-W;"█KOPF█",W;"█WAPPEN"
READY.
```

```
400-FACHER MUENZWURF


      ●=KOPF        O=WAPPEN
```

●OOOOO●●●●●OO●O●●O●●●●O●OOO●O●O●OOO●●OOO●●●O●OOO●O

●O●O●O●OOOOO●OO●OO●OO●OO●●●●OO●O●●OOOO●OOOO●●OOOO●

OO●OOOO●OOO●OOOO●●●●●O●OO●●OO●O●●●O●OOOO●OO●O●O●●●

●O●OOO●OOOOO●●O●O●●●●●O●O●●O●O●OOO●●●O●O●O●OO●●●●O

OO●OOO●●OO●O●●●●OO●●●●O●●OO●O●OO●OO●●●●●OO●●●OO●O●

OOO●●OO●●O●O●O●●●●●O●OOOOOOOO●●O●●●OO●●●O●●O●O●O●O●

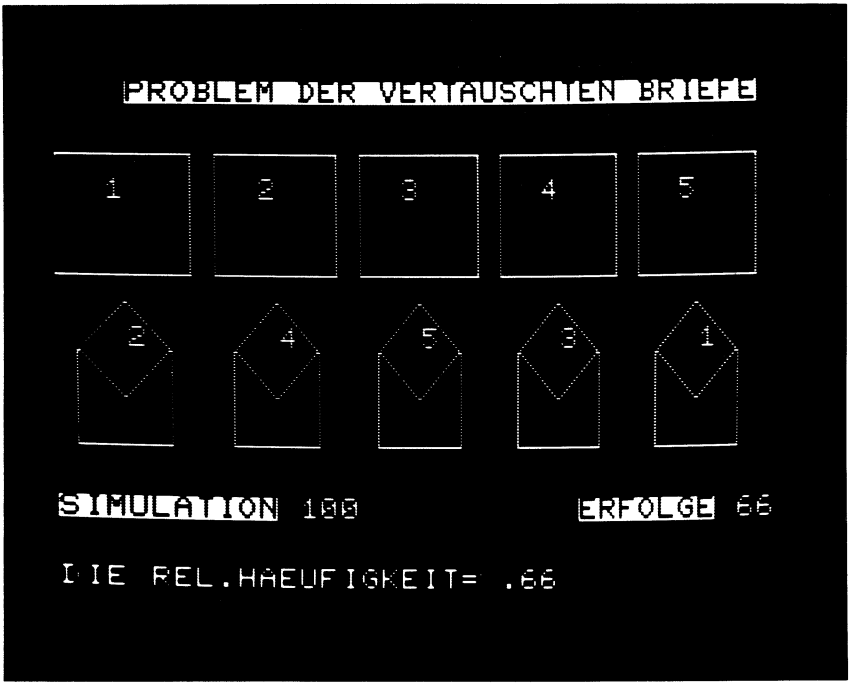
●OO●●●●O●O●●OOO●●●●●●OOO●●OOO●●●●OO●●●●●●OO●OO●●●O

O●O●OO●●OOO●●●●●●O●OOOOOO●●OO●●●●●OOO●●OOOOO●OOOO●

196 [illegible]   204 [illegible]

32. Vertauschte Briefe

Das Problem der vertauschten Briefe gibt es in vielen Abwandlungen. Jemand hat 5 Briefe geschrieben, diese werden zufällig in die vorbereiteten Umschläge gesteckt; mit welcher Wahrscheinlichkeit steckt wenigstens einer im richtigen Umschlag? Oder bei einer Party werden die Tanzpartner (alle verheiratet) ausgelost; mit welcher Wahrscheinlichkeit tanzt mindestens ein Ehepaar miteinander?



Zum folgenden Programm

Mit Hilfe von Zufallszahlen wird eine zufällige Anordnung (auch Permutation genannt) der Briefe ausgelost und geprüft, ob einer im richtigen Umschlag steckt. Dies ist der Fall, wenn eine Ziffer der Zufallspermutation am richtigen Platz steht. Entsprechend wird ein Zähler weitergezählt. Am Ende wird die relative Häufigkeit ausgedruckt. Die exakte Wahrscheinlichkeit beträgt

$$\frac{19}{30} = 0{,}633.$$

Bei einer größeren Anzahl von Versuchen wird diese Zahl ziemlich genau erreicht.

Interessant ist, daß bei einer sehr großen Zahl von Briefen die Wahrscheinlichkeit nicht mehr kleiner wird, obwohl man dies eigentlich erwartet. Sie beträgt

$$1 - \frac{1}{e} = 0{,}632,$$

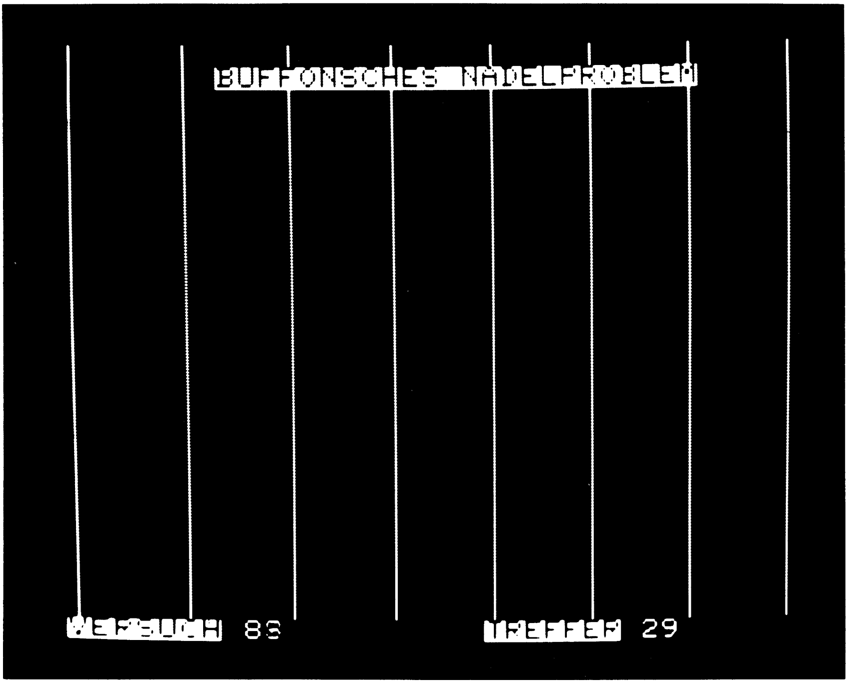
dabei ist e die berühmte Eulersche Zahl 2,7182818. . . .

```
100 REM PROBLEM DER VERTAUSCHTEN BRIEFE
110 :
120 REM IN 5 BESCHRIFTETE UMSCHLAEGE WERDEN ZUFAELLIG
130 REM DIE ENTSPRECHENDEN BRIEFE GESTECKT.
140 REM SIMULIERT WIRD DIE WAHRSCHEINLICHKEIT
150 REM,DASS MINDESTENS EIN BRIEF IM RICHTIGEN UMSCHLAG STECKT.
160 :
170 READ N : REM ANZAHL DER SIMULATIONEN
180 :
190 Z=0
200 GOSUB 430: REM GRAPHIK
210 :
220 REM ERZEUGUNG EINER ZUFALLSPERMUTATION
230 FOR J=1 TO N
240 FOR I=1 TO 5
250 L(I)=I
260 NEXT I
270 FOR K=1 TO 5
280 I=INT(5*RND(1)+1)
290 IF L(I)<0 THEN 280
300 P(K)=I
310 IF P(K)=K THEN Z=Z+1:GOTO 340
320 L(I)=-1
330 NEXT K
340 PRINT"[illegible]"
350 FOR L=1 TO 5
```

```
360 PRINTTAB(L*7-5)P(L);
370 NEXT L:PRINT
380 PRINT"████████SIMULATION█";J;TAB(25)"█ERFOLGE█";Z
390 NEXT J
400 :
410 PRINT"██";"DIE REL.HAEUFIGKEIT=";Z/N
420 END
430 PRINT"███████PROBLEM DER VERTAUSCHTEN BRIEFE██"
440 PRINT"┌──────┐ ┌──────┐ ┌──────┐ ┌──────┐ ┌──────┐"
450 PRINT"│  1   │ │  2   │ │  3   │ │  4   │ │  5   │ "
460 PRINT"│      │ │      │ │      │ │      │ │      │"
470 PRINT"│      │ │      │ │      │ │      │ │      │"
480 PRINT"└──────┘ └──────┘ └──────┘ └──────┘ └──────┘"
490 PRINT
500 PRINT"   /\       /\       /\       /\       /\    "
510 PRINT"  /  \     /  \     /  \     /  \     /  \   "
520 PRINT" |\  /|   |\  /|   |\  /|   |\  /|   |\  /|  "
530 PRINT" | \/ |   | \/ |   | \/ |   | \/ |   | \/ |  "
540 PRINT" |    |   |    |   |    |   |    |   |    |  "
550 PRINT" └────┘   └────┘   └────┘   └────┘   └────┘  "
560 RETURN
570 :
580 DATA 1000
READY.
```

33. Buffon-Nadelproblem

George-Louis Leclerc Buffon (1707 – 1788) war eigentlich Jurist, konnte sich aber nach einer Erbschaft ganz den Naturwissenschaften widmen. Auf Grund seiner Studien schrieb er eine 36-bändige Naturgeschichte "Histoire naturèlle générale et particulière".
Aus seinem "Essai d'arithmetique morale" stammt sein berühmter Nadelversuch:
Auf einem Tisch ist ein Parallelengitter vom Abstand d aufgemalt. Auf dieses Gitter werden zufällig Nadeln der Länge a geworfen.



Die Wahrscheinlichkeit, daß eine Nadel auf einer Parallelen zu liegen kommt, ist (siehe [6])

$$\frac{2 \cdot a}{\pi \cdot d} \qquad \text{mit } d > a$$

dabei ist π die berühmte Kreiszahl; d.h. das Verhältnis eines Kreisumfangs zu seinem Durchmesser.
Wirft man nun eine große Zahl von Nadeln, so kann die Wahrscheinlichkeit näherungsweise durch die entsprechende relative Trefferhäufigkeit ersetzt werden. Damit ist es möglich, die Zahl π, allerdings ungenau, zu berechnen.

Zum folgenden Programm

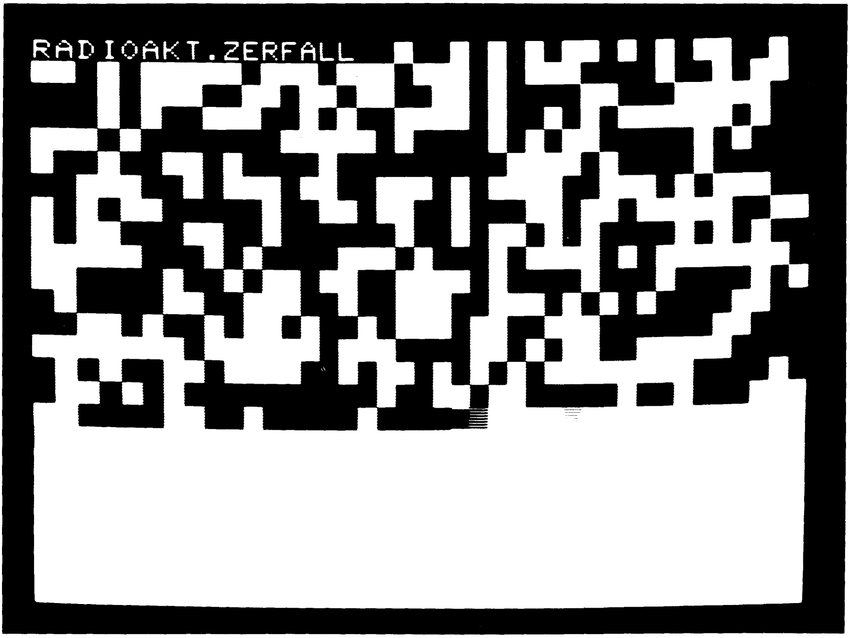
Im Programm wird der Nadelwurf durch Zufallszahlen simuliert und am Bildschirm graphisch dargestellt. Es wird die Trefferhäufigkeit gezählt und am Ende ein Schätzwert für die Zahl π ausgegeben.
Zur Vereinfachung wurde die Nadellänge auf den halben Gitterabstand festgelegt.

```
100 REM BUFFONSCHES NADELPROBLEM
110 :
120 INPUT"WIEVIELE VERSUCHE";N
130 PRINT""
140 FORI=1 TO 25
150 PRINT"|    |    |    |    |    |    |    |"
160 NEXT
170 PRINT"BUFFONSCHES NADELPROBLEM"
180 FOR K=1 TO N
190 X=RND(1):A=π*RND(2)
200 IF X<=.5*SIN(A) THEN 230
210 GOTO 290
220 :
230 REM TREFFER
240 T=T+1
250 PRINT"/ / / /";
260 FOR I=1 TO 100:NEXT I
270 PRINT"";"  ";"  ";"|";"  "
280 GOTO 340
290 REM NICHTTREFFER
300 PRINT"\ \ \ \"
310 FOR I=1 TO 100:NEXT I
320 PRINT"";"  ";"  ";"  ";"  "
330 :
```

```
340 PRINT"";
350 PRINT"VERSUCH";K;TAB(20)"TREFFER";T;"";
360 NEXT K
370 PRINT"SCHAETZWERT FUER π:";N/T
380 PRINT""
390 END
READY.
```

34. Radioaktiver Zerfall

Als physikalische Anwendung einer Monte-Carlo-Simulation soll der radioaktive Zerfall gezeigt werden.



Zum folgenden Programm

Die 1000 Bildschirmpunkte stellen die zerfallenden Atome dar. Da der Bildschirm erst mit Hilfe eines kleinen Maschinenprogramms revers geschaltet wurde, schalten die Punkte beim "Zerfall" auf Dunkel um. Die Lebensdauer der einzelnen Atome wird durch Zufallszahlen ausgelost. Dazu müssen die Zufallszahlen auf die sog. "Poisson-Verteilung" umgerechnet werden. Ist X eine gewöhnliche Zufallszahl, so stellt

$$-LOG(X)/LOG(2)$$

eine poisson-verteilte Zufallszahl mit dem Erwartungswert 1 dar. Diese Zufallszahlen stellen die Lebensdauern unserer Bildschirmpunkte dar.

Zuerst werden also die Atome zerfallen, die nur die 1. Halbwertszeit erleben, dann die, die in der 2. Halbwertszeit zerfallen und so fort. Da die Wahrscheinlichkeit für ein radioaktives Atom, eine Halbwertszeit zu überleben genau 1/2 ist, muß sich die Anzahl der überlebenden Atome bei jeder Halbwertszeit halbieren.
Nach 7 Halbwertszeiten sind $1 -(1/2)^7 = 99{,}2\ \%$ im Durchschnitt zerfallen.

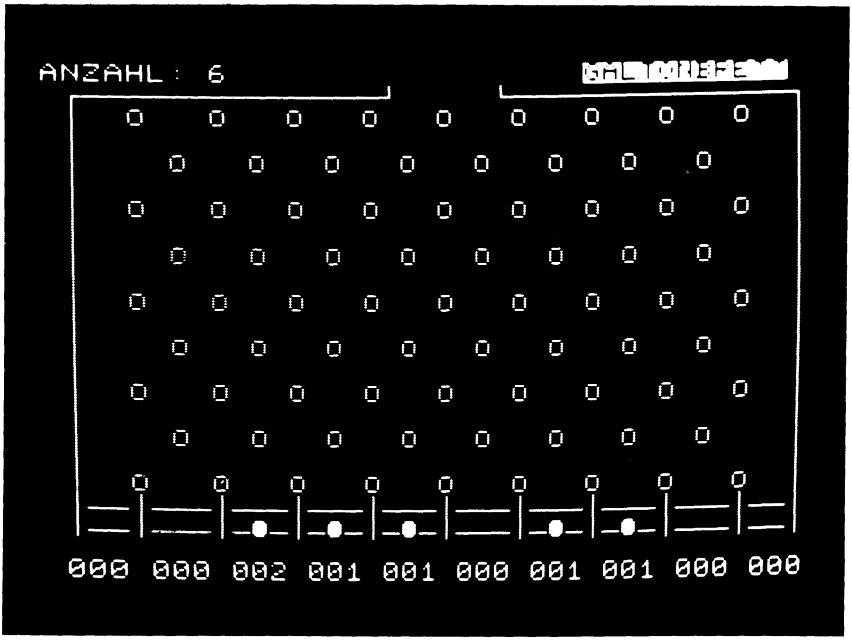
Dies läßt sich im Programmablauf gut verfolgen.

```
100 REM SIMULATION DES RADIOAKTIVEN ZERFALLS
110 :
120 FOR I=826 TO 863
130 READ A:POKE I,A :NEXT I
140 DATA 162,0,189,0
150 DATA 128,73,128,157
160 DATA 0,128,189,0
170 DATA 129,73,128,0
180 DATA 0,129,189,0
190 DATA 130,73,128,0
200 DATA 0,130,189,0
210 DATA 131,73,128,157
220 DATA 0,131,232,208
230 DATA 221,96
240 NEW
250 :
260 PRINT"[CLR]"
270 SYS 826
280 DIM A(1000)
290 B=LOG(2)
300 FOR I=32768 TO 32784:POKE I,32:NEXT
310 PRINT"[HOME]RADIOAKT.ZERFALL"
320 FOR I=17 TO 1000
330 X=RND(1):A(I)=-LOG(X)/B
340 IF A(I)<1 THEN POKE I+32767,32:A(I)=0
350 NEXT I
```

```
 360 FOR T=2 TO 10
 370 PRINT"■■";T-1;"■.HALBWERTSZEIT"
 380 FOR I=17 TO 1000
 390 IF A(I)=0 THEN 410
 400 IF A(I)< T THEN POKE I+32767,32:A(I)=0
 410 NEXT I
 420 NEXT T
READY.
```

35. Galtonbrett

Sir Francis Galton (1822 – 1911) war ein vielseitig interessierter Mann – er unternahm geographische Forschungsreisen, entwickelte die Vererbungslehre seines Vetters Charles Darwin weiter und führte die Fingerabdruck-Methode zur Personenidentifikation ein. So wundert es nicht, daß er sich auch mit der mathematischen Statistik befaßte. Bei diesen Studien erfand er das nach ihm benannte Galton-Brett:
Auf einem Brett sind mehrere Nagelreihen untereinander angeordnet. Läßt man Kugeln die Nagelreihen durchlaufen, so werden sie nach links oder rechts abgelenkt und fallen dann in Behälter am Fuß des Brettes. Durch Auszählen der Kugeln kann man die entsprechenden Wahrscheinlichkeiten näherungsweise experiementell bestimmen. Das Galtonbrett ist damit ein wichtiges Modell der sog. Binomialverteilung.



Zum folgenden Programm

Das Programm stellt das Galtonbrett graphisch mit 9 Nagelreihen dar. Das Fallen der Kugeln wird mit Hilfe von Zufallszahlen simuliert. Nach jedem Durchgang wird die Anzahl der in die einzelnen Behälter gefallenen Kugeln eingeblendet.

```
100 REM GALTONBRETT-SIMULATION
110 :
120 PRINT"■":INPUT"WIEVIELE KUGELN";N:PRINT"■"
130 PRINTSPC(28)"■■GALTONBRETT"
140 A$="  |──|───|───|───|───|───|───|───|───|──|"
150 A1$="■ 000 000 000 000 000 000 000 000 000 000"
160 B$="■■■■■■■■■■■■■■■■■■■■"
170 PRINT"■■  ┌─────────────┘      └─────────────┐"
180 FORI=32849 TO 33570 STEP 40:POKE I,93:POKE I+38,93:NEXT I
190 FOR I=1 TO 9:E=32768:M=9
200 IF I/2=INT(I/2) THEN E=E+2:M=8
210 FOR L=1 TO M
220 POKE E+I*80+L*4,87:NEXT L:NEXT I
230 PRINT B$;A$;A$;A1$
240 FOR I=1 TO N
250 D=0:E=32788:POKE E,81
260 GOSUB 490:POKE E,32
270 FOR K=1 TO 17 STEP 2
280 Z=2*INT(2*RND(K))-1
290 E=E+Z:D=D+INT(Z/2)+1
300 POKE E+K*40,81
310 GOSUB 490:POKE E+K*40,32
320 E=E+Z:POKE E+(K+1)*40,81
330 GOSUB 490:POKE E+(K+1)*40,32
340 NEXT K
350 POKE E+(K+1)*40,81
```

```
360 A(D)=A(D)+1
370 G=E+(K+1)*40+80
380 H=G+1:V=G-1
390 W=INT(A(D)/100)+48
400 B=INT(A(D)/10)-(W-48)*10+48
410 C= A(D)-INT(A(D)/10)*10+48
420 POKE V,W
430 POKE G,B:POKE H,C
440 PRINT"■";:PRINT"ANZAHL:";I
450 NEXT I
460 WAIT 152,1:POKE158,1:END
470 :
480 REM WARTESCHLEIFE
490 FOR J=1 TO 100:NEXT J:RETURN
READY.
```

36. Entwicklung einer Population

Simulation nennt man in den Naturwissenschaften das Nachahmen eines komplexen Vorgangs, der aus bestimmten Gründen nicht praktisch vollzogen werden kann. Meist ist es die Vielzahl, Verschiedenartigkeit oder ungenaue Kenntnis, die eine exakte Lösung des Problems verhindert.

Als Beispiel sei der 1972 erschienene Report "Grenzen des Wachstums" erwähnt, der damals großes Aufsehen erregte. Er faßte 99 so verschiedene Faktoren wie Rohstoffverbrauchs- und Umweltverschmutzungs-Rate, Nahrungsmittel- und Industrie-Produktion je Kopf zu einem sog. Weltmodell zusammen.
Dieses Simulationsmodell erlaubte die Prognose, daß es bei gleichbleibendem Zuwachs an Rohstoffverbrauch, Industrieproduktion und Bevölkerung nach der Jahrtausendwende zu einer ernsthaften Weltkrise kommen wird.

Bei den fünf folgenden Programmen handelt es sich um Simulationen aus dem Bereich der Populationsdynamik, Biologie, Medizin und Ökologie.

Für eine Population, die ungestört wächst, gilt folgendes Vermehrungsgesetz (vgl. [7])

$$N_{i+1} = (1+a-b \cdot N_i) \cdot N_i.$$

Dabei ist N_i die Anzahl der Individuen in einer Generation, entsprechend ist N_{i+1} die der folgenden Generation. a ist die Vermehrungsrate, b heißt der Gedränge-Koeffizient. Diese Konstanten können leicht aus drei aufeinander folgenden Generationszahlen berechnet werden. Für die USA gelten näherungsweise folgende Werte:

Bevölkerung 1890 : 63 Mill., a = 0,233, b = 0,0007

Mit diesen Angaben läßt sich die weitere Bevölkerungsentwicklung im Zehnjahresrhythmus berechnen. Wie man dem Programmausdruck entnimmt, ergibt die Hochrechnung für das Jahr 2000 eine Bevölkerung von 255 Mill.
Für Deutschland läßt sich keine derartige Rechnung aufmachen, da wegen der Verluste aus beiden Weltkriegen obige Populationsgleichung nicht anwendbar ist.

```
100 REM POPULATIONSGLEICHUNG
110 :
120 REM POPULATIONSGLEICHUNG DER FORM N=(1+A-B*N)*N
130 :
140 REM EINLESEN DER WERTE
150 READ N
160 READ A,B
170 DEF FNR(X)=INT(100*X+.5)/100
180 :
190 PRINTTAB(2)"USA-POPULATION"
200 PRINT "JAHR","MILL."
210 PRINT
220 FOR J=1890 TO 2000 STEP 10
230 PRINT J,FNR(N)
240 N=(1+A-B*N)*N : REM POPULATIONSGLEICHUNG
250 NEXT J
260 END
270 :
280 DATA 62.948
290 DATA .232912,6.71071E-4
READY.
```

POPULATIONSGLEICHUNG

USA-POPULATION

JAHR	MILL.
1890	62.95
1900	74.95
1910	88.64
1920	104.01
1930	120.98
1940	139.33
1950	158.75
1960	178.82
1970	199.01
1980	218.78
1990	237.62
2000	255.07

37. Füchse und Hasen

Leben zwei verschiedene Populationen nebeneinander, wobei die eine als Beute für die andere dient, werden die Populationsgleichungen als Räuber-Beute-Modell bezeichnet. Sind X_i, X_{i+1} die Populationszahlen der Beutetiere und Y_i, Y_{i+1} entsprechend die der Räuber, so gilt (vgl. [7])

$$X_{i+1} = (1 + a \cdot X_i - b \cdot X_i \cdot Y_i)$$

$$Y_{i+1} = (1 + c \cdot X_i \cdot Y_i - d \cdot Y_i).$$

a ist wieder die Vermehrungsrate der Beute, d ist Sterberate des Räubers. Die Koeffizienten b und c kennzeichnen die Verluste der Beutetiere bzw. die Zunahme der Räuber. Im Programm wurden die fiktiven Daten $X_i = 200$ Hasen, $Y_i = 20$ Füchse, a = 0,3, b = 0,01, c = 0,002, d = 0,5 benutzt.

Wie der Programmausdruck zeigt, steigt die Anzahl der Hasen zunächst an. Durch die vermehrte Beute können nun auch die Füchse mehr Jungtiere großziehen, d.h. auch die Zahl der Füchse wird größer. Wegen der angestiegenen Zahl von Räubern nimmt nun die Zahl der Hasen ab. Durch die verringerte Zahl von Beutetieren werden nun auch die Füchse weniger und die Hasen können sich verstärkt vermehren. Der Kreislauf der Natur beginnt von Neuem.

```
100 REM FUECHSE UND HASEN
110 :
120 REM     X ZAHL DER HASEN
130 REM     Y ZAHL DER FUECHSE
140 :
150 REM RAEUBER-BEUTE-GLEICHUNGEN:
160 REM DX/DT=A*X-B*X*Y
170 REM DY/DT=C*X*Y-D*Y
180 REM
190 :
200 REM EINLESEN DER KONSTANTEN
210 READ A,B,C,D
220 :
230 REM EINLESEN DER GENERATIONENZAHL
240 READ G
250 :
260 REM EINLESEN DER ANFANGSWERTE
270 READ X,Y
280 :
290 PRINT"[illegible]FUECHSE UND HASEN"
300 PRINT"[illegible]GENERATION HASEN FUECHSE"
310 PRINT" 0",X,Y
320 FOR T=1 TO G
330 X=X+(A*X-B*X*Y)
340 Y=Y+(C*X*Y-D*Y)
350 PRINT T,INT(X+.5),INT(Y+.5)
360 NEXT T
370 :
```

```
 380 DATA .3,.01,.002,.5
 390 DATA 20
 400 DATA 200,20
READY.
```

FUECHSE UND HASEN

```
.
GENERATION HASEN FUECHSE
 0          200       20
 1          220       19
 2          245       19
 3          273       19
 4          301       21
 5          327       25
 6          344       29
 7          346       35
 8          329       41
 9          294       44
 10         252       44
 11         216       41
 12         191       37
 13         179       31
 14         177       27
 15         182       23
 16         195       21
 17         213       19
 18         237       19
 19         264       19
 20         293       21
```

38. Ausbreitung einer Epidemie

Für die Ausbreitung einer Epidemie lassen sich folgende Gleichungen aufstellen:

Ist X, Y, Z die Zahl der Gesunden, Kranken und Immunen, so verringert sich die Zahl der Gesunden um die der Infizierten

$$a \cdot X \cdot Y$$

Gleichzeitig nimmt die Zahl der Immunen zu um

$$b \cdot Y$$

Die Zahl der Kranken ist gleich der Gesamtzahl N, vermindert um die Zahl der Gesunden und der Immunen (von Todesfällen wird abgesehen)

$$Y = N - X - Z$$

In das Programm wurden folgende Werte eingegeben: Anfangswerte für Gesunde 1990, für Kranke 10, a = 0,0005, b = 0,05.

Wie der Programmausdruck zeigt, steigt die Zahl der Kranken rapide an bis zum Höchststand 1675. Dies entspricht 83,7 % der Gesamtbevölkerung. Nach diesem Höchststand nimmt die Zahl der Kranken jedoch nur langsam ab. An diesem Zahlenbeispiel sieht man, welche verheerende Folgen eine Epidemie haben kann und wie wichtig eine Isolierung der Erkrankten ist.

```
100 REM EPIDEMIE
110 :
120 REM      X ZAHL DER GESUNDEN
130 REM      Y ZAHL DER KRANKEN
140 REM      Z ZAHL DER IMMUNEN
150 :
160 REM EPIDEMIE-GLEICHUNGEN :
170 REM   X=X-A*X*Y
180 REM   Z=Z+B*Y
190 REM   Y=N-X-Z
200 :
210 REM EINLESEN DER INFEKTIONS- U.IMMUNISIERUNGSRATE
220 READ A,B
230 :
240 REM EINLESEN DER ANFANGSWERTE
250 READ X,Y
260 :
270 REM EINLESEN DER ZEITINTERVALLE
280 READ P
290 :
300 Z=0
310 N=X+Y: REM GESAMTZAHL
320 :
330 PRINT"[illegible]EPIDEMIEVERLAUF[illegible]"
340 PRINT"[illegible]TAGE      KRANKE"
350 FOR T=1 TO P
```

```
 360 PRINTT,INT(Y)
 370 X=X-A*X*Y
 380 Z=Z+B*Y
 390 Y=N-X-Z
 400 NEXT T
 410 END
 420 :
 430 DATA .0005,.05
 440 DATA 1990,10
 450 DATA 20
READY.

EPIDEMIE

.
TAGE       KRANKE
 1           10
 2           19
 3           37
 4           72
 5           139
 6           261
 7           473
 8           805
 9           1225
 10          1583
```

11	1713
12	1675
13	1597
14	1519
15	1443
16	1371
17	1302
18	1237
19	1175
20	1116

39. Wirtschaftsmodell

Das Volkseinkommen Y_t einer Nation setzt sich zusammen aus den Konsumausgaben K_t der Privatverbraucher, den Investitionen I_t der Wirtschaft und den Staatsausgaben S_t:

$$Y_t = K_t + I_t + S_t.$$

Nach Samuelson (vgl. [9]) lassen sich folgende vereinfachende Annahmen machen:

(1) der Konsum K_t eines Jahres ist proportional zum Volkseinkommen des Vorjahres Y_{t-1} : $K_t = a \cdot Y_{t-1}$

(2) die Investitionen sind proportional zum Anstieg des Konsums

$$I_t = b \cdot (K_{t-1} - K_{t-2})$$

(3) die Staatsausgaben sind näherungsweise konstant

$$S_t = c.$$

Setzt man diese 3 Bedingungen in die obere Gleichung ein, so erhält man die Gleichung

$$Y_t = (a+ab) \cdot Y_{t-1} + ab \cdot Y_{t-2} + c.$$

Sie erlaubt aus dem Volkseinkommen zweier aufeinander folgenden Jahre die weitere Entwicklung zu berechnen.
Zur Vereinfachung wird im Programm das Volkseinkommen als Vielfaches der Staatsausgaben behandelt, c kann also 1 gesetzt werden.
Im Programm wurde mit $Y_o = 8$ und $Y_1 = 10$ ein Wirtschaftsaufschwung eingegeben. a wurde 0,9 und b = 1 gesetzt (fiktive Daten).
Wie der Programmausdruck ausweist, gibt es zunächst einen Aufschwung, der dann nach einigen Jahren von einer Rezession abgelöst wird.

```
100 REM SAMUELSON-WIRTSCHAFTSMODELL
110 :
120 REM Y(T) VOLKSEINKOMMEN
130 REM K(T) KONSUM
140 REM I(T) INVESTITIONEN
150 REM S(T) STAATSAUSGABEN
160 REM Y(T)=K(T)+I(T)+S(T)
170 :
180 REM EINLESEN DER PARAMETER A,B
190 READ A,B
200 REM EINLESEN DER ANFANGSWERTE Y0,Y1
210 READ Y0,Y1
220 PRINT"JAHR     VOLKSEINKOMMEN"
230 PRINT" 0",Y0:PRINT" 1",Y1
240 :
250 C=1 : REM STAATSAUSGABEN
260 FOR T=2 TO 15
270 Y2=(A+A*B)*Y1-A*B*Y0+C
280 PRINTT,INT(10*Y2+.5)/10
290 Y0=Y1:Y1=Y2
300 NEXT T
310 :
320 DATA .9,1
330 DATA 8,10
READY.
```

WIRTSCHAFTSMODELL

JAHR	VOLKSEINKOMMEN
0	8
1	10
2	11.8
3	13.2
4	14.2
5	14.7
6	14.6
7	14.1
8	13.2
9	12.1
10	10.9
11	9.7
12	8.7
13	7.9
14	7.4
15	7.2

40. Wasserverschmutzung

Das folgende Programm simuliert die Verschmutzung eines Sees, in den die Abwässer einer Stadt eingeleitet werden.

Folgende Daten werden benützt:

Wassermenge W des Sees 10 Mill. m^3
Verschmutzungsgrad am Anfang 0,5 %

Zufluß des Sees 50.000 m^3 pro Tag
Verschmutzungsgrad 0,3 %

Abfluß des Sees 50.000 m^3 pro Tag

Abwässer der Stadt 4.000 m^3 pro Tag
Verschmutzungsgrad 10 %

Das Programm berechnet nun die Anzahl der Tage bis der max. Verschmutzungsgrad 1 % des Sees erreicht worden ist.
Ergebnis: Bereits nach 366 Tagen ist unter den obengenannten Bedingungen der Verschmutzungsgrad 1 % erreicht.

```
100 REM WASSERVERSCHMUTZUNG
110 :
120 REM DIESES PROGRAMM SIMULIERT DIE WASSERVERSCHMUTZUNG
130 REM EINES SEES,DER VON EINEM ZUFLUSS UND EINER STADT
140 REM VERSCHMUTZT WIRD
150 REM ES WIRD BERECHNET,WANN EIN VORGEGEBENER VERSCHMUTZUNGSGRAD
160 REM ERREICHT WIRD.
170 :
180 REM W WASSER DES SEES IN M↑3
190 REM V1 VERSCHMUTZUNGSGRAD
200 READ W,V1
210 :
220 REM Z ZUFLUSS IN M↑3/TAG
230 REM V2 VERSCHMUTZUNGSGRAD
240 READ Z,V2
250 :
260 REM A ABFLUSS IN M↑3/TAG
270 READ A
280 :
290 REM K KANALISATIONSABWAESSER DER STADT IN M↑3/TAG
300 REM V3 VERSCHMUTZUNGSGRAD
310 READ K,V3
320 :
330 REM V VERSCHMUTZUNGSGRAD,BIS ZU DEM BERECHNET WERDEN SOLL
340 READ V
350 :
```

```
360 REM ANFANGSWERTE
370 T=0 : REM ZAEHLT DIE TAGE
380 S=W*V1:REM SCHMUTZMENGE AM ANFANG
390 :
400 PRINT"WASSERVERSCHMUTZUNG"
410 PRINT"TAGE    SCMUTZMENGE   VERSCHMUTZ.ANTEIL"
420 T=T+1
430 S=S+Z*V2+K*V3
440 S=S-(S/W)*A
450 PRINT T,INT(S+.5),INT(1E5*S/W+.5)/1E5
460 IF S/W > V THEN END
470 GOTO 420
480 :
490 DATA 1E7,.005
500 DATA 50000,.003
510 DATA 50000
520 DATA 4000,.1
530 DATA .01
READY.
```

SCHLÜSSEL-WÖRTER in Commodore-BASIC

ABS (absolute) gibt den Absolut-Betrag einer Zahl an

AND logische UND-Verknüpfung

ASC (ASCII-Character) ordnet jedem Zeichen die ASCII-Nummer zu (American Standard Code for Information Interchange)

CHR$ (character) ordnet jeder ASCII-Nummer das zugehörige Zeichen zu

CLR (clear) löscht alle Variablen und Dimensionierungen

CONT (continue) ermöglicht Fortsetzung des Programms nach Drücken der BREAK-Taste oder nach einem STOP-Befehl

DATA sind numerische und alphanumerische Variablenwerte, die innerhalb des Programms durch READ gelesen werden

DEFFN (define function) definiert eine Funktion, die mit Hilfe von FN. aufgerufen werden kann

DIM (dimension) reserviert Speicherplätze für Felder

END (end) stoppt die Programmausführung

FOR . . NEXT erzeugt eine Zähl-Wiederholungsanweisung (Schleife)

GET liest einzelne Zeichen von der Tastatur ein

GOSUB . . RETURN bewirkt Sprung in ein Unterprogramm und Rückkehr

GOTO (go to) Sprunganweisung

IF . . THEN Entscheidungs-Anweisung

INPUT ermöglicht Dateneingabe über Tastatur

INT (integer) rundet jede nicht-ganzzahlige Zahl auf die nächstkleinere ab

LEFT$ (left) trennt linken Teil einer alphanumerischen Variablen ab

LEN (length) gibt die Länge einer alphanumerischen Variablen an

LOG (logarithm) natürliche Logarithmus-Funktion zur Basis e=2.71828183 ...

MID$ trennt Teile einer alphanumerischen Variablen heraus

NEW löscht das Programm im Speicher

NOT logische Verneinung

ON . . GOSUB berechneter Sprung ins Unterprogramm

ON . . GOTO berechneter Sprung innerhalb des Programms

OR logische ODER-Verknüpfung

PEEK (peek) liest den Inhalt eines Speicherplatzes

POKE (poke) dient zur Beschreibung von Speicherplätzen

PRINT (print) ist Ausgabe-Anweisung

READ (read) siehe DATA

REM (remark) beginnt eine Kommentar-Zeile

RESTORE (restore) ermöglicht, daß DATA-Werte erneut gelesen werden können

RETURN (return) siehe GOSUB bzw. ON . . GOSUB

RIGHT$ (right) trennt den rechten Teil einer alphanumerischen Variablen ab

RND (random digit) erzeugt Zufallszahl

SGN (sign) gibt das Vorzeichen einer Zahl an

SQR (square root) liefert die Quadratwurzel einer Zahl

STEP (step) liefert Schrittweite der FOR . . NEXT-Schleife

STOP (stop) programmierbarer Halt innerhalb des Programms

STR$ (string) wandelt Zahl in eine alphanumerische Variable um

SYS ruft Maschinen-Programm auf

TAB Tabulator-Funktion

VAL (value) wandelt alphanumerische Variable in eine Zahl um

WAIT (wait) Anweisung zum Warten, bis eine bestimmte Taste gedrückt oder bestimmter Speicherplatz ungleich Null wird

LITERATURVERZEICHNIS

[1] Ahl D.H.: BASIC computer games, Morris Town 1978

[2] Ahl D.H.: More BASIC Computer Games, Morris Town 1979

[3] Baumann R.: BASIC-Aufgabensammlung, Stuttgart 1981

[4] Gardner M.: Mathematische Rätsel und Probleme, Braunschweig 1966

[5] Gardner M.: Further Mathematical Diversions, Gretna 1977

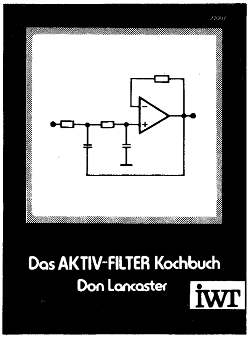
[6] Haber H.: Das Mathematische Kabinett Folge 1, München 1973

[7] Hadeler K.P.: Mathematik für Biologen, Heidelberg 1974

[8] Menzel K.: BASIC in 100 Beispielen, Stuttgart 1981

[9] Tischel G.: Angewandte Mathematik, Frankfurt 1980

[10] Van Delft P. / Botermans J.: Denkspiele der Welt, München 1979



Das AKTIV-FILTER-Kochbuch

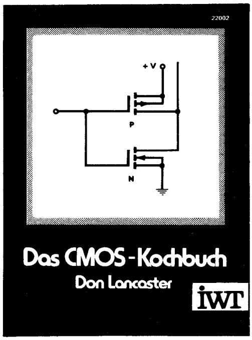
Von Don Lancaster, Übersetzung aus dem Amerikanischen,
ca. 270 Seiten, mit zahlreichen Abbildungen, 1982,
ISBN 3-88322-007-8, geb., DM 48.–

Das AKTIV-FILTER-Kochbuch stellt ein praktisches, anwenderorientiertes Nachschlagewerk für jeden dar, der etwas über den Aufbau eines speziellen Filters wissen will. Der Name "Lancaster", auch Autor des CMOS-Kochbuches, steht für die Qualität dieses Werkes.

Das CMOS-Kochbuch

Von Don Lancaster, Übersetzung aus dem Amerikanischen, ca. 420 Seiten, mit zahlreichen Abbildungen, 1980,
ISBN 3-88322-002-7, geb. DM 48.--

Die digitale CMOS-Bausteinserie ist eine der modernsten und zukunftsichersten Logikfamilien. Hier liegt nun die erste und umfassendste neutrale Darstellung zu dieser modernen Technologie vor. Für alle Interessenten bringt "Das CMOS-Kochbuch" eine Fülle von wertvollen Informationen, die es sehr schnell zu einem unentbehrlichen Ratgeber für jeden Elektroniker machen werden.





Das CMOS-Taschenbuch, Band 1, Standardbausteine

ca. 230 Seiten, mit zahlreichen Abbildungen, 1981,
ISBN 3-88322-003-5, kart. , DM 32.--

Dieses Taschenbuch ist die ideale Ergänzung zum CMOS-Kochbuch. Es bietet eine übersichtliche Zusammenstellung der Standardtypen aller integrierten CMOS-Bausteine. Die Erfassung aller namhafter Hersteller sichert eine entsprechende Vollständigkeit.

In Vorbereitung: Band 2, Spezialbausteine. Inhalt:
Industrielle Steuerbausteine, Zeitgeber- und Uhrenschaltungen, Zähler, Mikroprozessoren, Mikroprozessor-Hilfsbausteine und Speicher.
2. Halbjahr 1982, ca. 250 Seiten, **ISBN 3-88322-009-4, kart.**

TTL-Taschenbuch, Teil 1 und 2

Das TTL-Taschenbuch bietet eine klar gegliederte Zusammenstellung aller gängigen TTL-Bausteine der namhaften Hersteller. Es sind alle aktuellen TTL-Familien, wie Standard-TTL, Low-Power-TTL, Schottky-TTL, Low-Power-Schottky TTL, High-Speed-TTL, und Fast-Schottky-TTL erfaßt.

Wertvolle und unentbehrliche Informationen sind: Anschlußbild für die Pinbelegung, Inhaltsbeschreibung der Bausteine, Signale oder Pegel pro Anschluß, Ansteuerung und Signalabgabe, Anwendungsmöglichkeiten, wichtige Daten in Kurzform, abschließend Type und Name des Bausteins.

Hingegen werden die immer gleichen Informationen, die die TTL-Serie betreffen, nur einmal am Anfang des Buches aufgelistet, um die vielen Zusatzinformationen übersichtlich aufführen zu können.

Da nicht jeder Hersteller alle Bausteine produziert, wird diese Produktinformation am Schluß des Teils 2 in Tabellenform zusammengefaßt.

TTL-Taschenbuch, Teil 1 1982, ca. 250 Seiten,
ISBN 3-88322-008-6, kart. , ca. DM 32.--

Teil 2, ISBN 3-88322-010-8,
In Vorbereitung, Ende 1982.

I wie Information • W wie Wissenschaft • T wie Technologie

Der APPLE-Software Wegweiser '82

Eine Auswahl deutschsprachiger Programme und wer sie liefert

Mit einem Vorwort von Dipl.-Ing. Günther Daubach, Leverkusen,
ca. 150 Seiten, 1982, **ISBN 3-88322-012-4, kart., DM 32,–**

Der APPLE-Software Wegweiser bietet erstmals die Möglichkeit, gezielt nach den verschiedenen Kriterien: Programme, Hersteller, Sach- und Fachgruppe spezielle Programme auszuwählen. Ein ausführliches Register und ein alphabetisches Händlerverzeichnis runden den "Wegweiser" ab. Er soll jährlich einmal — auf den neuesten Stand gebracht — erscheinen.





CBM 8050 - DOS - Listing

Betriebssystem im Detail

Von Dr. Ruprecht, München, ca. 168 Seiten, zahlreiche Programme,
ISBN 3-88322-015-9, Ringordner, DM 104.--

In einer Art überdimensionalen Kreuzworträtsel ist es dem Autor gelungen, das Betriebssystem der CBM 8050 zu "knacken". Dieses Werk ist nicht nur für "tätige" Programmierer, sondern auch für "nur neugierige" Computer-Fans: Sie bekommen hier Einblick in das Zusammenwirken von zwei 6502-Prozessoren. Hinzu kommt der Datenverkehr über PIAs mit dem Rechner. Dieses System arbeitet dann sehr selbständig zeitlich parallel. Ein Leckerbissen für alle, die ihren Commodore noch besser nutzen wollen.

CP/M für die Praxis, Band 1:

Vom Umgang mit CP/M - Eine allgemeinverständliche Einführung

Von Bernd Pol, Stuttgart, ca. 480 Seiten mit zahlreichen praktischen Beispielen, 1982, **ISBN 3-88322-004-3, geb., DM 48.—**

Am Anfang dieser auf zunächst 8 Bände angelegten Reihe über das neue Betriebssystem CP/M steht eine allgemeinverständliche Einführung. Dem Charakter eines solchen Buches gemäß ist es experimentierend angelegt und führt den Leser im ständigen Kontakt mit dem Computer Schritt für Schritt zu einer umfassenden Übersicht bis hin zur Beherrschung des Systems auch bei Fehlfunktionen.

In Vorbereitung: Band 2: Vom Umgang mit CP/M-86 - Eine allgemein verständliche Einführung, ISBN 3-88322-005-1
Band 3: CP/M im Einsatz - Tips und Tricks für die Programmierung. ISBN 3-88322-006-X



Herrmanns CBM Programmsammlung in Basic

Band 1: Spiele, Knobeleien und Simulationen

Von Dietmar Herrmann, Anzing, ca. 240 Seiten,
in Vorbereitung 1982
ISBN 3-88322-013-2, kart., DM 32,–

Der erste Band einer neuen Reihe, die mit zahlreichen Programmen für Spiele und "ernsthafte" Themen den Commodore-Computer dem Benutzer näherbringt. Dietmar Herrmann hat aus seiner Schulpraxis heraus Programme entwickelt, die das Lernen und Spielen mit dem Computer zum Vergnügen machen.

Weitere Bände in dieser Reihe, die im Laufe dieses Jahres folgen sollen:
Band 2: Wirtschaft; Band 3: Mathematik

TRS-80 Assembler-Programmierung

Von Dipl.-Ing. Günther Daubach, Leverkusen,
ca. 180 Seiten, in Vorbereitung 1982,
ISBN 3-88322-017-5, kart., ca. DM 48,–

Für den Anfänger eine verständliche Einführung in die "Muttersprache" des TRS-80 mit zahlreichen Erklärungen der Befehle an den Z80 Mikroprozessor, für den Kenner der Assemblerprogrammierung eine Erweiterung seines Wissens über Details, wie z.B. die Verwendung nützlicher ROM-Routinen und die Speicherung von Variablen, Datenaufzeichnung auf Cassette und Diskette, sowie auch die Unterschiede zwischen TRS-80, Modell I und III.

I W T Verlag GmbH

Dahlienstraße 4, 8011 Vaterstetten, Telefon (08106) 4986

32.-